Sound Synthesis from Shape-Changing Geometric Models

by

Cynthia Maxwell

B.S. (Rensselaer Polytechnic Institute) 1995
M.S. (Stanford University) 1997
M.S. (University of California at Berkeley) 2005

A dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION

of the

UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Carlo Séquin, Chair
Ruzena Bajcsy
David Wessel

Fall 2008

The dissertation of Cynthia Maxwell is approved.

Chair	Date
	Date
	Date

University of California, Berkeley

Fall 2008

Sound Synthesis from Shape-Changing Geometric Models

by

Cynthia Maxwell

Abstract

Sound Synthesis from Shape-Changing Geometric Models

by

Cynthia Maxwell

Doctor of Philosophy in Computer Science

University of California, Berkeley

Carlo Séquin, Chair

This dissertation is about computational tools to aid in the design of musical instruments using arbitrary geometric models as a basis. In this work we develop numerical methods and software systems that simulate interacting with geometric shapes to synthesize sound.

We propose a method for rapidly estimating modal parameters of a shape by using the modal information from similar shapes. In this way, we can rapidly compute the updated modal parameters of a geometric model as the shape is changing – a phenomenon not currently modeled in other sound synthesis systems.

These techniques provide a new framework for interactive instrument design and create a platform for interacting with novel virtual instruments. We implement these techniques

in an interactive software system to demonstrate the efficiency and compelling interactivity of this new mode of sound synthesis.

Carlo Séquin
Dissertation Committee Chair

To all the people who made sacrifices so I could explore my scientific curiosity.

Contents

List of Figures

List of Tables

Acknowledgements

Special thanks to my husband Justin Maxwell for his patience and understanding. Thanks also to my family and to all of my friends, particularly David Bruyns and my teachers at Bernal Yoga.

Thanks to all of the people who shared their knowledge of audio techniques: Doug Wyatt, James McCartney, Bill Stewart, Frank Baumgarte, John Lazzaro, David Wessel, Adrian Freed, Julius Smith, and Antonie Chaigne.

Thanks to the engineering professors and post doctoral researchers who coached me through my research: Carlo Séquin, Ruzena Bajcsy, Panayiotis Papadopoulos, Robert Taylor, Ushnish Basu, and Fei Ma.

Thanks to the math professors, postdoctoral researchers and Ph.D. students for their wisdom: David Bindel, Jim Demmel, Beresford Parlett, and Johnathan Dorfman.

Thanks to Apple Incorporated for creating such an energetic and collaborative environment for me to engage in. And to UC Berkeley for having such open and collaborative departments. This thesis would not have been possible without their cooperation.

Thanks to the NASA engineers who inspired me to study mechanical engineering. And to the Ohio Areospace Institute and the NASA Scholar program for making it possible for me to attend Rensselaer.

Finally, thanks to the teachers of Holland High School who sparked my interest in math and science.

Chapter 1

Introduction

In the last decade, many researchers have used modal synthesis for sound generation. The modal method naturally lends itself to sound synthesis, because it allows one to model object sounds without the need to explicitly program the properties of several resonators. Instead, resonator properties are determined by the system of equations representing the object's motion. Using a modal decomposition, one can convert this system of equations into simple, independent differential equations in one variable, which is much more efficient than solving the original coupled system. The modal response of an object is determined by performing a system eigendecomposition to arrive at eigenvectors forming a basis for the object's motion and eigenvalues determining the resonant frequencies of the object. Together, the eigenvalues and eigenvectors provide the information needed to recreate the object's surface as it deforms. Because the object's modal response is typically computed before interactive sound synthesis, the software used to compute the representation of object motion can simply perform a projection of the interaction forces onto the modes of the system. This determines which modes are activated, which in turn determines how an object will behave when forces are applied to it. In this way, object simulation at run-time

is reduced to matrix-vector multiplication. Modeling such dynamics without the use of modal synthesis would otherwise require integrating a set of coupled second-order differential equations with explicit constraints.

For large systems, such as those obtained from finite-element analysis of musical instruments, the initial modal decomposition is time-consuming. A limitation of using the modal method for "design" is therefore the cost of computing the eigen-information. This affects the modifications that can be made to the object's geometry and material after the eigensolution has been determined. To design instruments from physical simulation however, one would like to be able to compute modes in real-time, so that the geometry, and therefore the spectrum, of an instrument can be changed interactively.

A summary of previous work in sound synthesis using physical models is given in report of Bruyns, 2006 [21]. In that report, the aim was to explore the simulation of thin plate models for sound synthesis and evaluate the accuracy of the simulation by comparison with recordings. This previous report also described a system for interactive generation of sound from geometric models. However the work was limited in the sense that:

- It only modeled the resonator response to a simple impulse function. When striking an object, contact can actually involve several micro-bounces creating intricate contact profiles which were not captured by this system.
- The sound synthesis algorithm assumed a far-field listening scenario and neglected the object's interaction with the surrounding acoustic medium. For realistic rendering of sound from arbitrary positions near to the object's surface, the previous simple point source model is not valid.
- The models analyzed by our previous system were static, however we would like to

be able to make changes to the models and audition the associated change in timbre. Using traditional modal synthesis methods, we would need to compute a new modal decomposition with each change in the geometry, making the analysis too slow for interactive use.

The main contributions of this thesis address these previous limitations and extends our sound synthesis system by[1]:

- Using an approximation of arbitrary forcing functions for musically interesting interactions. Using this approximation we can apply a simulated transducer load to our models allowing us to simulate a plate reverberation system.
- Using methods for fast analysis of changing geometries, we now allow for interactive instrument design and the creation of a shape-changing virtual instrument within a sound synthesis environment.

1.1 Assumptions

In the real world, object motion can be nonlinear due to the material of the object, the amplitude of the vibration, and the method of excitation. Nonlinear behavior breaks the assumptions of the modeling schemes discussed in previous research, thus we can not use the usual tools, such as linear superposition to solve for object motion. Nonlinear behavior invalidates the use of linear modal analysis as described in Chapter 2.1.

Fletcher [36] refers to instruments the primary action of which is within the nonlinear domain as "essentially nonlinear", and instruments, such as the percussive ones treated

[1]Preliminary modeling of fluid-structure interaction and an implementation in a sound rendering environment was presented in Bruyns,2007 [22]. We consider this to be a separate problem to instrument design and thus will not be discussed in depth in this thesis.

in this thesis, as "incidentally nonlinear". This distinction is made because essentially nonlinear instruments produce harmonic tones as a result of their nonlinearity. Incidentally nonlinear instruments, however, produce inharmonic sounds and behave linearly, but can exhibit nonlinearities if the level of impulsive excitation is large.

Methods for extending linear modal superposition to the nonlinear domain include nonlinear normal modes, single-mode methods, invariant manifold systems, harmonic-balance and nonlinear modal analysis.

The most attractive of these methods, nonlinear modal analysis, commonly involves model reduction, such as described in Chapter 6, and subsequent modal decomposition and differentiation of the linear normal modes to form a nonlinear modal basis. In this way, the techniques we present for rapidly analyzing a system after shape modification are still useful when modeling large deformation in a nonlinear modal synthesis setting.

The examples we present in this thesis assume only small deformation.

1.2 Outline

The rest of this thesis is organized as follows:

- In Chapter 2 we setup the mathematical background for analyzing a system of equations resulting from the finite element method. The result of this analysis is a set of uncoupled single degree-of-freedom resonators. After describing techniques for transforming the system into this form, we present techniques for solving the single-degree-of-freedom systems under common loading conditions. Finally, we present a method for approximating the solution of these systems for arbitrary forcing functions.

- In Chapter 3 we review the methods for synthesizing sound using the mathematics presented in Chapter 2 as a basis.

- In Chapter 4 we review the validation of this method using simple and complex geometries.

- In Chapter 5 we present alternative methods for analysis of shape-changing axisymmetric objects. The goal here is to solve for the modal parameters in an efficient manner by exploiting axisymmetry.

- In Chapter 6 we focus on methods for fast approximation of the modal parameters for arbitrarily shaped objects after moderate changes are made. In this chapter we find that it is possible to accurately predict the frequency spectrum of one shape by using the modal basis from nearby shapes.

- In Chapter 7 we demonstrate using these techniques in a software instrument system and describe a new shape-changing virtual instrument.

- In Chapter 8 we present results of using approximate solutions for the motion of resonators under arbitrary loading conditions, using plate reverberation as an example. We also describe a method for making our analysis usable in different audio rendering systems that can be linked with new tactile controllers.

- In Chapter 9 we present the conclusions to this work.

Chapter 2

Mathematical Preliminaries

2.1 Modal Analysis

For a system with n degrees-of-freedom (DOFs), the governing equations of motion are a set of n coupled ordinary differential equations of second order. The solution of these equations becomes complicated when the size of the system is large or when the forcing functions of the system are non-periodic. In such cases, it is convenient to express the deformation of the object as linear combinations of normal modes of the system. Such a transformation uncouples the equations of motion into a set of n independent scalar differential equations.

The canonical system of equations resulting from discretization using the finite element method is as follows:

$$M\ddot{u} + C\dot{u} + Ku = f(t) \tag{2.1}$$

where M is the matrix representing the distribution of mass in the system, C is a measure of damping, and K is the stiffness matrix. This equation expresses the balance of forces generated by the acceleration, velocity and displacement of the object. In this form the

equations are coupled, and thus the solution involves manipulation of these large system matrices. Alternatively, modal analysis seeks to decouple this system into many single degree-of-freedom (SDOF) oscillators.

Without damping, the procedure for uncoupling these equations is straightforward using the general eigenvalue decomposition $Kx = \lambda Mx$. However, with damping, decoupling these equations requires some assumptions to be made [83].

In many finite element representations, there is no straightforward method of generating the damping matrix C explicitly. "A major reason for this is that, in contrast with inertia and stiffness forces, the physics behind the damping forces is in general not clear. As a consequence, modelling of damping from the first principles is difficult, if not impossible, for real-life engineering structures. The common approach is to use the proportional damping model, where it is assumed that the damping matrix is proportional to mass and stiffness matrices [3]." This proportionality is represented as:

$$C = \alpha_1 M + \alpha_2 K \tag{2.2}$$

where $\alpha_1$and α_2 are real scalars. This damping model is also known as Rayleigh damping or classical damping. Modes of classically damped systems preserve the simplicity of the real normal modes as in the undamped case.

Currently, there are no lookup tables for determining the correct values for α_1 and α_2 for a given material. These parameters are usually determined by optimally finding the values that closely approximate a measured damping matrix C. Since the measurement of the matrices M and K are usually robust, Rayleigh damping can be used with some confidence. In this way, α_1 and α_2 are usually solved for using a least squares minimization of error to some measured system behavior.

"The main limitation of the proportional damping approximation comes from the fact that the variation of damping factors with respect to vibration frequency cannot be modelled accurately by using this approach. Experimental results however suggest that damping factors can vary with frequency [3]."

As a result, other researchers have looked for expressions that generalize the proportional damping assumption. One such method specifies damping ratios, ζ, for any number of modes and the damping matrix is then represented as a Caughey series:

$$c = m \sum_{b=0}^{N-1} a_b [m^{-1} b]^b \tag{2.3}$$

After specifying the damping ratios, one then solves for the coefficients, a_b, using the relation:

$$\zeta_n = \frac{1}{2} \sum_{b=0}^{J-1} a_b \omega_n^{2b-1} \tag{2.4}$$

However, a difficulty arises because the algebraic expressions represented in Equation 2.4 are ill-conditioned due to the large differences in ω_n. Moreover, when specifying several terms in the Caughey series, the resulting d matrix will not retain the sparsity of the original stiffness and mass matrices. Given the added computation needed when analyzing large dense systems, this method is rarely used [85], [25].

Another method for specifying modal damping is the orthogonal matrix method. This method expresses the damping matrix as a sum of a series of matrices, each of which specifies damping for a particular mode. In this way, each matrix is not ill-conditioned as in the Caughey series method [85].

Other researchers [34] have looked for a method of diagonalizing the system by extending the eigendecomposition procedure to include diagonalization of the matrix C. In this

procedure, one looks for a set of vectors that diagonalize K and M and nearly diagonalizes C (leaving small entries off the diagonal, which are then discarded). As with many assumptions, careful analysis of the effect of discarding these off-diagonal terms should be performed, as there are examples in which the errors introduced by this method are counter-intuitive [41]. That is, errors due to the decoupling approximation can increase continuously while the modal damping matrix becomes more and more diagonal with its off-diagonal elements decreasing in magnitude continuously. One would assume however that the more diagonal the matrix, the smaller the errors induced by the decoupling approximation, but this is not strictly the case as described by Ma, 1993 [41].

Research into the error introduced by assuming proportional damping is ongoing, and current results seem to suggest that there may never be one static assumption that accurately diagonalizes a coupled damped system [54].

In our sound synthesis engine, we allow the user to change α_1 and α_2 for the desired sound effect, and for now, we will assume proportional damping as in Equation 2.2. Substituting this expression back into Equation 2.1, we obtain:

$$M(\ddot{u}+\alpha_1\dot{u})+K(\alpha_2\dot{u}+u) \quad = \quad f(t) \tag{2.5}$$

We assume a particular solution of the form:

$$u \quad = \quad Zv \tag{2.6}$$

where Z is the matrix of eigenvectors that diagonalizes the system. Substituting back into Equation 2.5 and pre-multiplying by Z^T we obtain:

$$Z^TMZ(\ddot{v}+\alpha_1\dot{v})+Z^TKZ(\alpha_2\dot{v}+v) \quad = \quad Z^Tf(t) \tag{2.7}$$

Using the definitions $Z^TMZ=I$ and $Z^TKZ=\Lambda$, where Λ is the diagonal matrix of eigenvalues,

this equation simplifies to:

$$\ddot{v} + (\alpha_1 + \alpha_2 \omega^2)\dot{v} + \omega^2 v = Z^T f(t) \tag{2.8}$$

where $\lambda = \omega^2$.

Equation 2.8 represents the uncoupled bank of resonators, oscillating at the natural frequencies determined by the eigenvalues of the system. By solving each equation for u we can represent the response at any location on the object at any time. Therefore, to solve for the motion at a particular point on the object we weight the contributions of the various modes on the spatial positions of interest.

2.1.1 Storage

In general the system matrices will be sparse due to the compact support of the finite element interpolating functions. For example, Figure 2.1 shows the non-zero entries in the stiffness matrix for a plate with 1700 DOFS.

Given this sparse structure, we can optimize storage by using use the Compressed Column format which reduces the storage from $O(n^2)$ to $O(n)$ [12]. This format ignores the zero entries, storing only those entries that need to be used for computation.

After decomposition, however, the eigenvector matrix is full and storage is again $O(mn)$ where m is the number of eigenvalues sought and n is the number of DOFs. This is the dominant storage requirement as the eigenvector matrix must be stored in RAM for interactive sound rendering. For example, if we assume that we have computed 1000 modes for a geometry with 1000 DOFS, and we wish to utilize all of the modes to generate sound, then we would need to store roughly 4 MB of data.

This requirement might seem significant, however it is generally held that structure

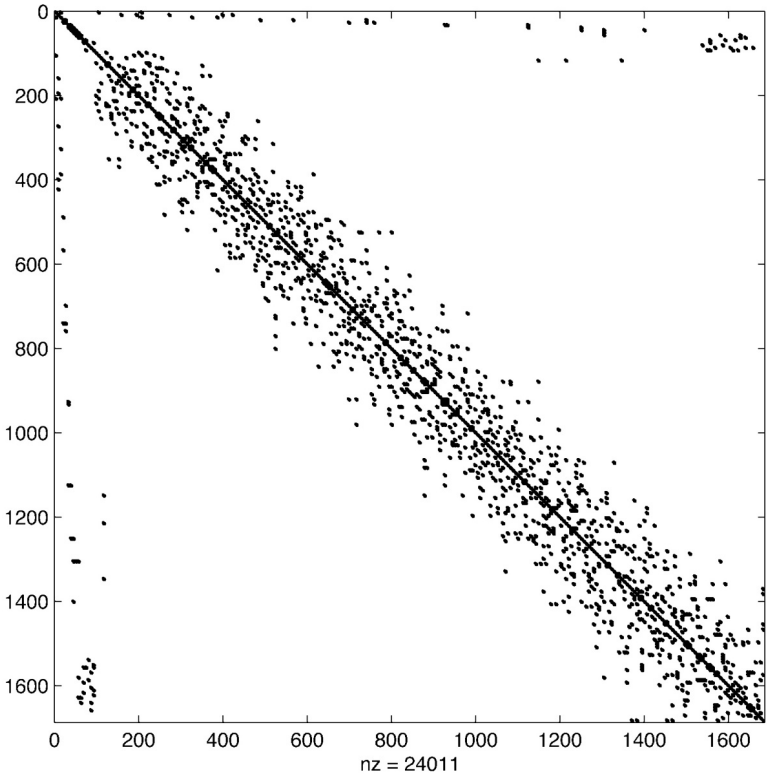

Figure 2.1. Matrix structure for a square plate model.

motion can be accurately simulated using only a few of the lower normal modes [25]. In this way, it is not necessary to use all of the modes when generating sound. In fact, many researchers utilizing the modal analysis method for real-time applications prune the modes of interest based on the range of motion expected in the simulation. The effect of modal truncation is an ongoing area of research.

For time-harmonic forcing functions, it may be trivial to determine which modes will be important in capturing the object motion. For more complex loading, one might first need to analyze the input in the spectral domain to determine which resonant frequencies could be activated. In general, the contributions of each mode for a given loading scenario and boundary conditions should be evaluated when deciding whether to prune the mode or not.

An interesting study by van den Doel et. al. [78] describes a significant increase in performance when synthesizing sound using only those modes that contribute to the final perceived sound in a psychoacoustic sense. That is, by searching for modes that are likely to be masked by other modes, one can prune the actual modes that are rendered in real time.

In our sound synthesis environment, we choose to retain all of the modes and let the user specify how many are used for computation. For fast analysis after shape modification, we use the methods described in Chapter 5 and Chapter 6 which use a projection onto a subset of eigenvectors thus decreasing the needed storage, while still retaining the modes of interest. This projection method has been in use for decades, and there are many successful examples of using the modal subspace technique in the engineering literature.

2.1.2 Solvers

Finite element matrices are symmetric because they represent a conservative system, and they are sparse because their entries are related through mostly local element connectivity. With proper node numbering, the matrices can also exhibit diagonal banding, which can expedite decomposition.

The mass matrix for structural problems is real, symmetric and positive-definite, and can be factored using Cholesky factorization [30]. If one uses the lumped mass model, then Cholesky factorization is made even easier because the mass matrix will be diagonal. Similarly, using a Galerkin's weighted residuals method, the stiffness matrix will also be symmetric positive-definite. For the examples in this thesis, we exploit these properties in matrix structure and utilize the direct methods in CLAPACK [4] to solve the general eigenvalue problem $Kx = \lambda Mx$.

2.1.3 Analysis

Strang, 1973 [76] and Meirtovitch, 1997 [57] both state that the approximation $\lambda_r^{(n)}$ to an eigenvalue is bounded by the following relation:

$$\lambda_r \leq \lambda_r^{(n)} \leq \lambda_r + ch^{2(k-p)}(\lambda_r^{(n)})^{k/p}, r = 1, 2, ..., n \tag{2.9}$$

where λ_r is the actual eigenvalue, n is the number of elements, h is the longest side of the finite element mesh, c is a constant, k is the degree of the interpolation functions, and p is the order of the highest spatial derivative. This means that the error in the approximation of the eigenvalues decreases with smaller element sizes (more elements used) and the rate of convergence increases with higher order interpolation functions.

However, Equation 2.9 also shows that as eigenvalues increase in value, the error also

increases. This means that higher frequencies are predicted less accurately. Generally less than one half of the approximated eigenvalues computed by the finite element method can be considered reliable. Because of this, one should include at least twice as many degrees of freedom in the model as the number of accurate modes desired [58]. The error in the higher modal frequencies further motivates mode pruning before simulation.

2.2 Resonators

Modal analysis involves computing the natural frequencies and mode shapes of a structure in order to determine how the structure moves when forced [38]. The modal method naturally lends itself to the synthesis of sounds produced by physical objects, because it allows one to model object sounds using resonator properties computed automatically from a physical model, rather than by direct programming.

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 and transform the solution from modal coordinates.

By representing the object's motion in this way we arrive at an analytic expression for the position of each surface point of the object at any time. By using an analytic representation, the evaluation of the position does not require numerical integration and thus is unconditionally stable. This is beneficial as the stability of the solution will not be sensitive to time-step size as is the case for typical temporal integration schemes. In this

chapter we present methods for solving for the response of resonators to various input forces and discuss methods for dealing with complex interaction methods in a sound synthesis environment.

2.2.1 Damping

Damping is generally defined as any means of dissipating some fraction of energy which is added to the system by the exciting forces. Passive damping, such as linear viscoelastic damping, is manifested as energy dissipation in the form of heat. In many polymeric materials damping occurs as a result of molecular interaction during cyclic deformation [43]. This form of thermoplastic damping occurs as the temperature of the parts of the body under compression is raised, while the parts of the body under tension is lowered. The resulting temperature gradient causes heat to flow, turning mechanical energy into thermal energy that is lost from the system.

In different materials, damping is caused by other factors: in amorphous crystals, deformation causes molecular rearrangements; in composite materials, deformation involves slip at material interfaces; in porous materials, fluid (such as air or water) flow causes damping. [32].

Damping can be characterized as the amount of energy that is lost from the system. Table 2.1 shows the loss factors for a few common materials when resonating at $1Hz$ as listed in Djoharian, 2001 [32]. This table shows that certain materials lose more energy than others for the same frequency of vibration. The differing levels of energy loss results in the material characteristics in the sounds each material generates.

Material	Loss Factor
Steel	0.0005
Aluminum	0.001
Glass	0.0043
Wood	0.02
Plexiglass	0.1

Table 2.1. Energy loss factors for different materials.

Our previous publication examines exterior damping caused by fluid-loading on a vibrating object [22]. In this chapter we assume damping is due to internal factors.

2.2.2 Perception

A large amount of research has tried to determine the importance of frequency and damping in our ability to characterize sound sources [50], [32]. Studies show that the sound signature of a material is a characteristic function of frequency and damping. Inertia and elasticity combine into one single variable, the speed of sound in a material, while the shape of an object can determine its reduced spectrum (frequency spectrum up to a scaling factor) independent of material.

It is also interesting to note that there are two wave speeds in homogenous solids – one for compression waves, and one for shear waves. For ideal materials, one can still obtain characteristic frequency scaling under geometric scaling provided that the ratio between these two wave speeds remains constant from one material to the next.

Studies have been performed that qualify a sound based on the decay of vibration. These studies find that for different degrees of energy loss (the number of cycles it took to decay to $1/10^{th}$ the original amplitude) the sound produced took on a different characteristic. Table 2.2 is a summary of these findings [51]. This means that for the materials in Table 2.1, increasing loss factors coincide with a decrease in discernible pitch.

Cycles	Sound
≥739	"Clang": approaching pure tone
73	"Bong": clear pitch
7.3	"Bunk": discernible pitch
1.5	"Thud": almost without pitch

Table 2.2. Perceived quality of sound based on the decay of the vibration as reported in Lakes,1998.

Other researchers have confirmed that listeners use the decay more than the frequency of a sound to determine the material of an object [47], [80]. All of these studies suggest that accurate determination of the decay parameters of a material will be essential in creating realistic instrument sounds. Moreover, they highlight that the damping assumption used in Chapter 2.1 will affect the quality of the simulation of a given object.

2.2.3 Linear Damped SDOF Resonator, Free Vibration

An acoustic instrument is divided into two parts: the exciter and the resonator. Even though the exciter is also a vibrating structure, the shape and material features have their greatest relevance in the resonators [32]. In most instruments, the resonator's vibrations are of small amplitude [32], and therefore, one can assume linearity.

In this section we will only consider passively-damped resonators, resonators whose damping comes from internal forces. Using these assumptions, the single degree-of-freedom (SDOF) system we are trying to solve is represented as:

$$m\ddot{x} + c\dot{x} + kx = 0 \tag{2.10}$$

A common form of Equation 2.10 is to divide by m and define three new terms:

$$\ddot{x} + 2\xi\omega_n\dot{x} + \omega_n^2 x = 0 \tag{2.11}$$

$$\xi = \frac{c}{c_{cr}} \tag{2.12}$$

$$c_{cr} = 2m\omega_n \tag{2.13}$$

$$\omega_n^2 = k/m \tag{2.14}$$

The ω_n is the damped circular natural frequency (in rad/sec), the dimensionless quantity ξ is called the viscous damping factor, and c_{cr} is called the critical damping coefficient [59]. The viscous damping factor is a measure of the severity of the damping.

If we assume a homogeneous solution of the form:

$$x = Ae^{\lambda t} \tag{2.15}$$

and substitute it back into Equation 2.10, assuming a solution valid for all t, we find the characteristic equation [27]:

$$\lambda^2 + 2\xi\omega_n\lambda + \omega_n^2 = 0 \tag{2.16}$$

Which gives us the roots:

$$\lambda_1 = \omega_n(-\xi + \sqrt{\xi^2 - 1}) \tag{2.17}$$

$$\lambda_2 = \omega_n(-\xi - \sqrt{\xi^2 - 1}) \tag{2.18}$$

$$\tag{2.19}$$

Because the radicand $(\xi^2 - 1)$ can be positive, negative, or zero, there are three types of damped motion: overdamped, critically damped, and underdamped.

Overdamped, $\xi > 1$

$$x = A_1 e^{\lambda_1 t} + A_2 e^{\lambda_2 t} \tag{2.20}$$

In this type of motion, x approaches zero as time grows and there is no oscillation.

Critically Damped, $\xi = 1$

$$x = A_1 e^{-\omega_n t} + A_2 t e^{-\omega_n t} \tag{2.21}$$

For this type of motion, there will also be no oscillation, and the system will reach equilibrium faster than an overdamped system.

Underdamped, $\xi < 1$

$$x = C e^{-\xi \omega_n t} \sin(\omega_d t + \psi) \tag{2.22}$$

$$\omega_d = \omega_n \sqrt{(1 - \xi^2)} \tag{2.23}$$

In this case, oscillations will arise that represent an exponentially decreasing harmonic function. The frequency ω_d is called the damped natural frequency and the period of oscillation is given by $\tau_d = 2\pi/\omega_d$.

Figure 2.2 shows the solution to Equation 2.16 for each type of damping.

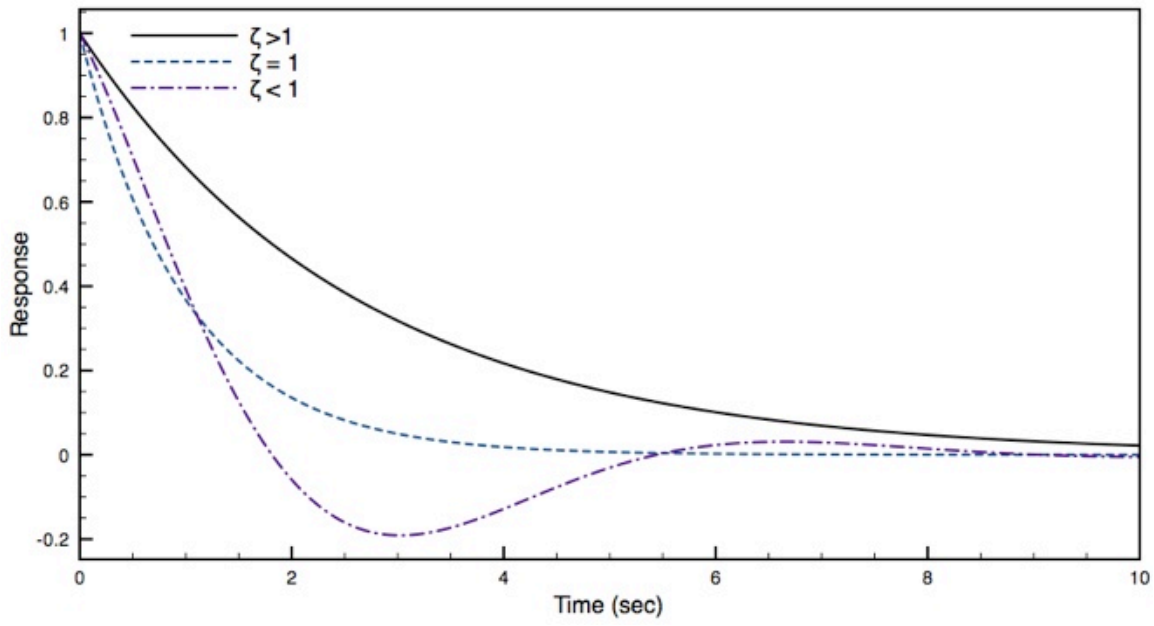

Figure 2.2. Response of a SDOF system to a unit impulse with various levels of damping.

Constants

The constants A_1, A_2, C and ϕ in the previous expressions are determined from the initial conditions of the system. For example, for the critically damped system, if we assume zero initial displacement and initial velocity proportional to the applied force, we have:

$$x(0) = A_1 + A_2 = 0 \tag{2.24}$$

$$\dot{x}(0) = p \tag{2.25}$$

which gives:

$$A_1 = \frac{p}{\lambda_1 - \lambda_2} \tag{2.26}$$

$$A_2 = \frac{p}{\lambda_2 - \lambda_1} \tag{2.27}$$

2.2.4 Linear Damped SDOF Resonator, Forced Vibration

In the previous section, we solved for free resonator vibration. For forced vibration, the system in Equation 2.10 becomes:

$$m\ddot{x} + c\dot{x} + kx = f(t) \tag{2.28}$$

Similar to Equation 2.11 we have:

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = F(t) \tag{2.29}$$

where $F(t) = f(t)/m$.

One method of solving this equation is to take its Laplace transform so that:

$$L[x(t)] = \bar{x}(s) \tag{2.30}$$

$$L[\dot{x}(t)] = s\bar{x}(s) - x(0) \tag{2.31}$$

$$L[\ddot{x}(t)] = s^2\bar{x}(s) - sx(0) - \dot{\bar{x}}(0) \tag{2.32}$$

$$L[F(t)] = \bar{F}(s) \tag{2.33}$$

Plugging these values into Equation 2.29 we find:

$$\bar{x} = \frac{1}{s^2 + 2\xi\omega_n s + \omega_n^2}[\bar{F}(s) + (s + 2\xi\omega_n)x_0 + \dot{x_0}] \tag{2.34}$$

By taking the inverse Laplace transform we can solve for displacement as

$$x(t) = \int_0^t F(\tau)h(t-\tau)d\tau \; + \; g(t)x_0 + h(t)\dot{x_0} \tag{2.35}$$

where

$$g(t) = e^{-\xi\omega_n t}\left(\cos(\omega_d t) \; + \; \frac{\xi\omega_n}{\omega_d}\sin(\omega_d t)\right) \tag{2.36}$$

$$h(t) = \frac{1}{\omega_d}e^{-\xi\omega_n t}\sin(\omega_d t) \tag{2.37}$$

The first term in Equation 2.35 is known as the convolution integral or Duhamel's integral. The second and third terms are called the transients because of the presence of $e^{-\xi\omega_n t}$ which is a decaying function of time [27], [70].

This method of solving for resonator motion is useful when applying forcing functions with known Laplace transforms. The following subsections demonstrate the solution using the Laplace transform method for such forcing functions.

Impulse

For example, the response to a unit impulse with zero initial conditions can be obtained by substituting $f(t) = F\delta(t)$ into the right hand side of Equation 2.28.

$$m\ddot{x} + c\dot{x} + kx = F\delta(t) \tag{2.38}$$

Then by taking the Laplace transform of both sides we have:

$$\bar{x} = \frac{F/m}{s^2 + 2\xi\omega_n s + \omega_n^2} \tag{2.39}$$

And taking the inverse Laplace transform we arrive at the solution for motion versus time:

$$x(t) = \frac{F}{m\omega_d} e^{-\xi\omega_n t} \sin(\omega_d t) \tag{2.40}$$

Harmonic Function

Similarly, if the system is excited by harmonic excitation, $f(t) = F\sin(\omega t)$, and the steady state solution is:

$$x(t) = \frac{F\sin(\omega t - \phi)}{\sqrt{(k - m\omega^2)^2 + (c\omega)^2}} \tag{2.41}$$

In fact, using this method the response to any periodic function can be obtained by expressing the excitation as linear combinations of harmonic functions using a Fourier series.

Step Function

For unit step function loading, we have the following response:

$$x(t) = (F/k) - (F/k)e^{-\xi\omega_n t}\left[\frac{\xi}{\sqrt{1-\xi^2}}\sin(\omega_d t) + \cos(\omega_d t)\right] \tag{2.42}$$

Ramp Loading

The response of the same system to a ramp load can be obtained by integrating the step response. In other words:

$$x(t) = (F/k\omega_n)(\omega_n t - \sin(\omega_n t)) \tag{2.43}$$

Figure 2.3 compares the response of a SDOF system, with stiffness $k = 4$, mass $m = 1$, damping $\xi = 0.1$, and force $F = 1$, to each of these types of loading.

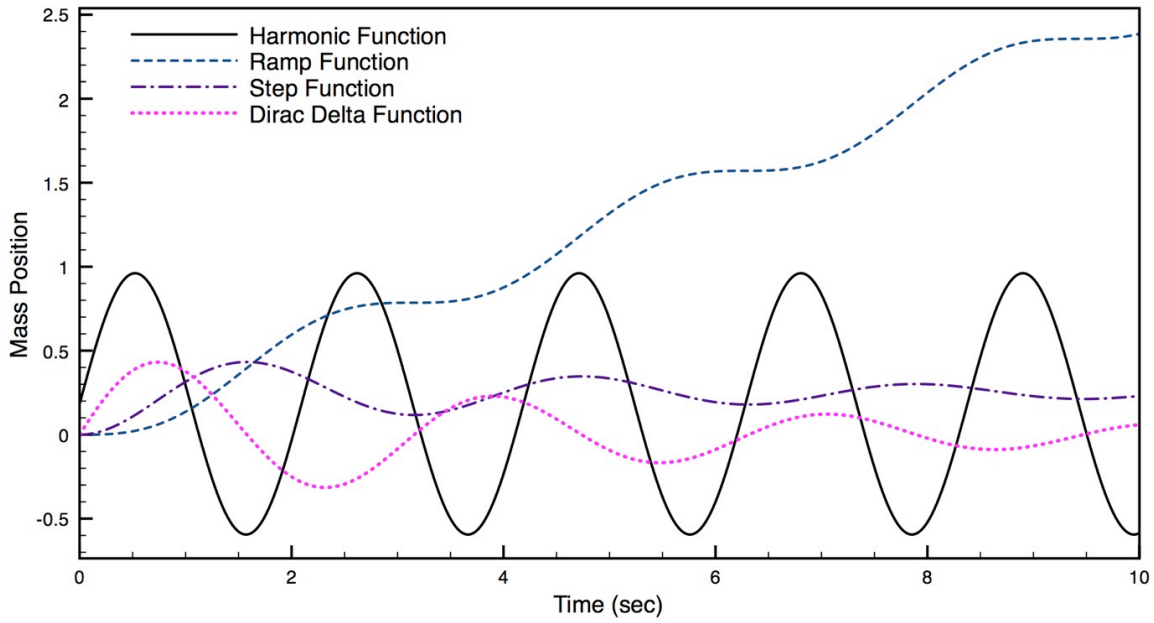

Figure 2.3. Mass-spring-damper response to varying forcing functions.

Arbitrary Loads

When using arbitrary loading, finding these forward and inverse Laplace transforms can be difficult. Other researchers have looked for closed-form expressions of such complex loading [24]. Others fall back on numerical integration techniques as described in Section 3.2.1. However, we know that any force-profile can be represented as a succession simple impulses [43] and thus the total response of a system at time t is the sum of all the responses to the succession of simple impulses. This response then can be represented by the convolution integral:

$$x(t) = \frac{1}{m\omega_d}\int_0^t f(\tau)\sin[(t-\tau)\omega_d]e^{-\xi\omega_n(t-\tau)}H(t-\tau)d\tau \tag{2.44}$$

where $H(t)$ is the Heaviside step function.

This means we can represent a discrete time excitation as a combination of delta functions:

$$f(n) = \sum_{k=0}^{\infty} f(k)\delta(n-k) \tag{2.45}$$

The response to this discrete-time excitation is then:

$$x(n) = \sum_{k=0}^{\infty} f(k)g(n-k) = \sum_{k=0}^{n} f(k)g(n-k) \tag{2.46}$$

which essentially approximates the response of a linear time-invariant discrete system in the form of a convolution sum, the discrete counterpart of the convolution or Duhamel's integral. This operation is essentially evaluating a numerical quadrature rule for the convolution integral.

Similar to the research of DiFilippo and Pai [31] and Rath [71, 72], we can use this representation to simulate interactions that are musically interesting. For example, we

can consider contact sounds such as impact, bouncing, sliding and rolling. Each of these interactions defines a forcing function for which it would be difficult to find forward and inverse Laplace transforms.

2.2.5 Forced Vibration Optimized

Instead of evaluating the discrete convolution integral as in Equation 2.46, DiFilippo and Pai [31] use an alternate, computationally efficient technique to generate the response to non-harmonic excitation. They use a recursive method based on scaling the response at the previous time step, $y_n(k-1)$, and adding that to the force applied, $F(k)$, at the new time. Thus, for each resonator:

$$y_n(0) = a_n F(0) \tag{2.47}$$

$$y_n(k) = \exp\left(i\frac{\Omega_n}{F_s}\right) y_n(k-1) + a_n F(k) \tag{2.48}$$

where F_s is the sampling frequency, and

$$\Omega_n = \omega_n + id_n \tag{2.49}$$

$$d_n = f\pi\tan(\phi) \tag{2.50}$$

where f is the natural frequency of vibration and ϕ is an internal damping factor.

Thus the overall resonator response generated at time t is:

$$s(t) = Re\left(\sum y_n(k)\right) \tag{2.51}$$

This method is actually just a numerical integration rule for a scalar first-order constant-coefficient ODE with forcing.

The efficiency of the recursive method comes from modeling the response to the previous impulses as one term with an exponential decaying scale factor. A non-recursive method,

on the other hand, would require one to evaulate the response of the resonator to excitation at each of previous time steps as in Equation 2.46. This means at each time step t, one would evaluate $t \times SampleRate \times n$ equations. Using the recursive method we only need to evaluate n equations.

2.2.6 Summary

In this chapter we examined the effect of applying different forcing functions to SDOF resonators. By approximating a complex forcing profile as a series of impulse responses, we can now use arbitrary resonator excitation mechanisms in the modal synthesis engine. In Chapter 8, we utilize these techniques and to simulate transducer loading on a synthetic plate.

Chapter 3

Sound Rendering

3.1 Background

When rendering sound from a discretized model, the aim is to treat each element on the surface like a vibrating piston [33], [46]. Each piston can be considered to be a source pushing on the surrounding air with attenuation according the orientation, distance, and visibility to the listener. The aggregate of these attenuated sound waves (by superposition) is the final tone perceived.

For two-dimensional (2D) models (which only represent the object mid-surface), we use the top face of these models for the piston computation (simulating a baffled source). This approximation is valid since we do not compute sound wave reflections with the environment. Three-dimensional (3D) objects, on the other hand, simulate omnidirectional sources.

Because we do not compute a detailed fluid-surface interaction, the final sound rendered simulates a far-field listening scenario. Preliminary research described in Bruyns, 2007 [22] investigates modeling additional acoustic radiation phenomena. In this chapter, we describe simple omnidirectional synthesis.

3.2 Methods

Computing the sound generated by an object in response to a particular striking force vector requires solving Equation 2.1. In the previous chapter we described how to solve this equation after modal decomposition. That is, we solve Equation 2.40 for each SDOF system separately. Then by transforming the system from modal coordinates and summing the amplitudes of these SDOF resonators together, we arrive at the final waveform for a given audio sample. In the previous chapter we also described the solution of SDOF resonators to different applied forces. We use these solutions to compute a simulation of object motion. The time to perform this accumulation depends on the number of resonators used.

3.2.1 Numerical Integration

Another means of solving systems with arbitrary force profiles is to use direct numerical integration. While this method solves the system as a large coupled algebraic expression, it is useful when the boundary conditions, material properties, or other important features of the model can change with time. This method is also useful when a representation of the system in the frequency domain is not feasible [56].

There are several different numerical integration techniques providing differing amounts of stability and efficiency. The most commonly used methods are Direct Integration Methods [14].

In this scheme, one is solving the canonical second order partial differential equation

$$M\ddot{x} + C\dot{x} + Kx = f(t) \tag{3.1}$$

where capital letters indicate that the mass, damping and stiffness terms represent matrices. These matrices can result from a finite element or finite difference discretization.

The Newmark scheme solves this equation as:

$$x(t+\Delta t) = F_{eff}/K_{eff} \tag{3.2}$$

where

$$\begin{aligned}
K_{eff} &= a_0 M + a_1 C + K \\
F_{eff} &= F(t+\Delta t) + M(a_0 x(t) + a_2 \dot{x}(t) + a_e \ddot{x}(t)) + C(a_1 x(t) + a_4 \dot{x}(t) + a_5 \ddot{x}(t)) \\
a_0 &= \frac{1}{\alpha \Delta t^2} && (3.3) \\
a_1 &= \frac{\delta}{\alpha \Delta t} && (3.4) \\
a_2 &= \frac{1}{\alpha \Delta t} && (3.5) \\
a_3 &= \frac{1}{2\alpha} - 1 && (3.6) \\
a_4 &= \frac{\delta}{\alpha} - 1 && (3.7) \\
a_5 &= \frac{\Delta t}{2}(\frac{\delta}{\alpha} - 2) && (3.8)
\end{aligned}$$

The secondary variables $\dot{x}$ and $\ddot{x}$ are updated as:

$$\begin{aligned}
\ddot{x}(t+\Delta t) &= a_0(x(t+\Delta t) - x(t)) - a_2 \dot{x}(t) - a_3 \ddot{x}(t) && (3.9) \\
\dot{x}(t+\Delta t) &= \dot{x}(t) + a_6 \ddot{x}(t) + a_7 \ddot{x}(t+\Delta t) && (3.10)
\end{aligned}$$

where

$$\begin{aligned}
a_6 &= \Delta t(1-\delta) && (3.11) \\
a_7 &= \delta \Delta t && (3.12)
\end{aligned}$$

If we assume proportional damping then we can approximate the damping matrix, C, from Equation 2.2.

We can compare how well the Newmark integration method approximates the closed form solution given in Equation 2.40 for a unit impulse load. For example, we solve for the response of a SDOF resonator with $k = 100$, $m = 1$, $\alpha_1 = 1 \times 10^{-7}$ and $\alpha_2 = 0.1$. Starting with a sample rate of 10 Hz, we see a poor fit to the actual solution (Figure 3.1). For a sample rate of 100 Hz, the fit is much better (Figure 3.2).

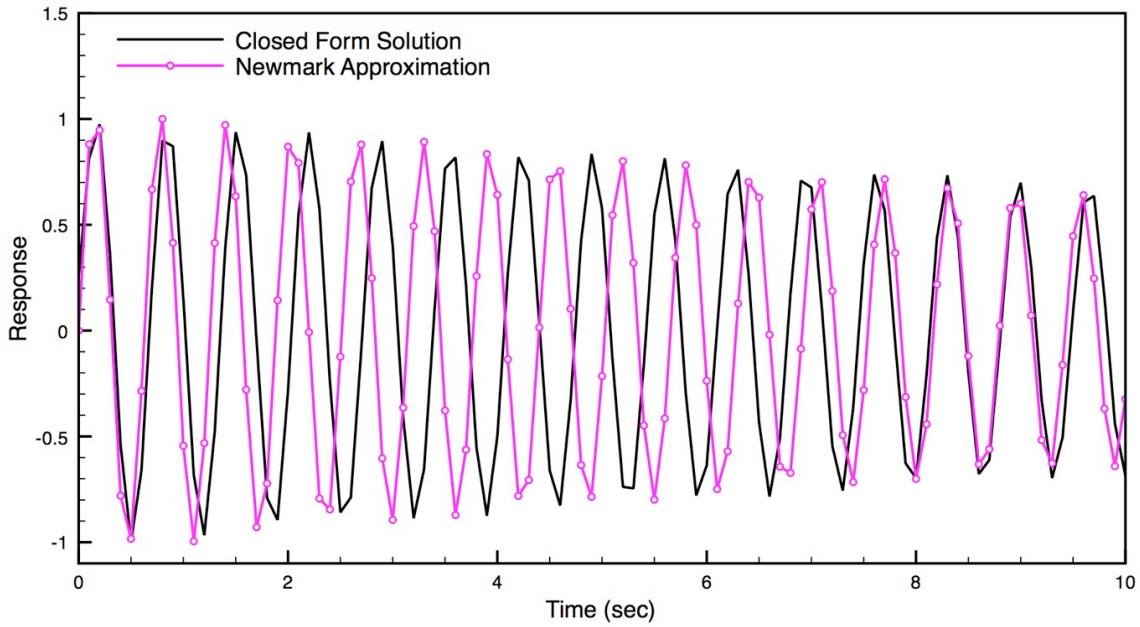

Figure 3.1. Response comparison $\omega_n = 10$ Hz, $\alpha_1 = 1 \times 10^{-7}$ and $\alpha_2 = 0.1$. Sampling frequency = 10 Hz.

One of the drawbacks of numerical integration however is the need to solve the system $Ax = b$ (Equation 3.2) at each time step. This process can cost as much as $O(n^3)$ when using normal Gaussian elimination, where n is the size of the system. As a result, alternate methods such as iterative techniques, multi-grid and model reduction are used.

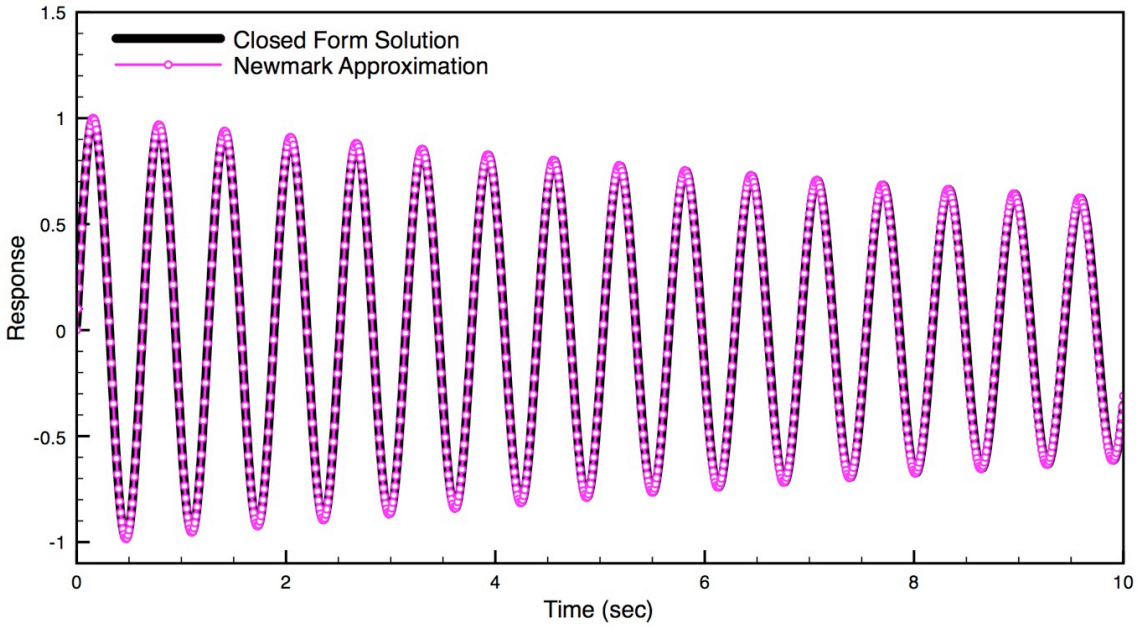

Figure 3.2. Response comparison $\omega_n = 10$ Hz, $\alpha_1 = 1 \times 10^{-7}$ and $\alpha_2 = 0.1$. Sampling frequency = 100 Hz.

3.3 Discussion

Due to the relative high cost of per-sample evaluation using numerical integration, we instead choose to use the modal analysis method for real-time sound synthesis. Modal analysis requires an expensive setup phase before the rendering engine starts. After this initial cost, however, evaluating the solution per-sample is extremely efficient.

In this way, once the modal parameters are found, sound rendering can begin. Chapters 5 and Chapter 6 will explain methods for rapidly computing or approximating the modal parameters after the object's shape has changed. Using these methods, we can use the new modal parameters directly in the sound rendering engine for an updated representation of the timbre of an object. In this way, we only temporarily stop synthesizing sound instead of waiting much longer for a full decomposition.

Chapter 4

Model Validation

In order to compare the frequency spectrum computed from the 3D models to the frequency spectrum from actual objects, we performed a series of tests. In the first tests, we modeled square plates made from steel and from aluminum and compared our simulation results to recordings of the actual physical objects. Next we chose two more complex shapes made from aluminum and steel and compared their frequency spectra.

The following sections describe the experimental setup, an analysis of the audio recordings, the model computations and an evaluation of their comparison.

4.1 Experimental Setup

For the experimental setup, we used an acoustically dampened room for recording (courtesy Apple Incorporated). This reduced the sound reflections from the room walls. In this way, we hoped to capture only sound waves traveling directly from the vibrating plate to the microphone. The simple shapes were suspended by a smooth string through a small hole at the center that allowed the plate to hang freely, balanced and level. This mounting was

used to provide an approximation of a free surface. It should be noted that by suspending the object from the center, we suppress the activation of the modes that have non-zero amplitudes at that point. We assume that this suppression effect is small because we used a soft, thin rope that allowed transverse motion. Similar suppression can be attributed to gravity, but we hoped that the effect would be negligible. The object was suspended from a rigid aluminum frame to provide a fixed support for the setup. We chose an open frame to minimize reflections off of the frame. Figure 4.1 demonstrates the experimental setup.

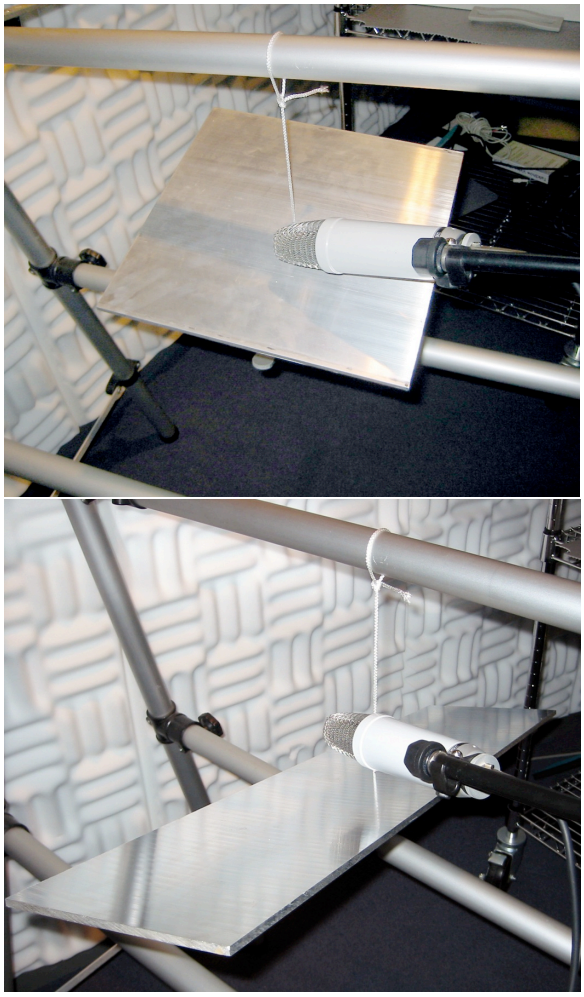

Figure 4.1. Aluminum square and rectangular plates, recording setup.

We suspended the complex shapes by hanging a smooth rope around a balanced point

in the form and then suspended them from the aluminum frame. Figure 4.2 demonstrates this setup. This method of suspension may have suppressed some of modes affecting the top of the shape.

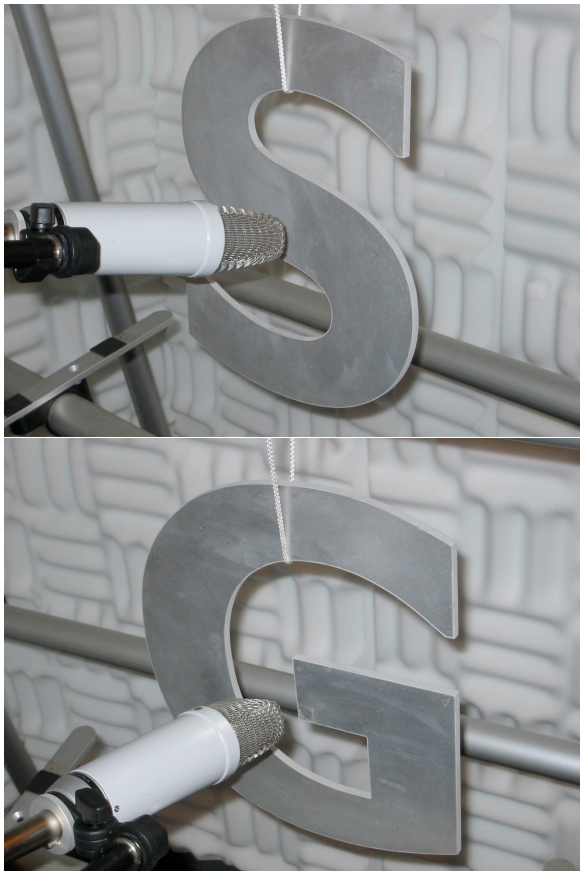

Figure 4.2. Complex shape recording setup.

4.2 Object Strike

Each object was struck with a 1-foot long, 1/8th-inch wide metal rod. The object was struck normal to the surface at a location 1 inch in from the mid-edge of the object, as indicated in Figure 4.3. For the complex shapes, the mid-edge was approximated. The force used to strike the object did not produce any large motion or rotations of the object.

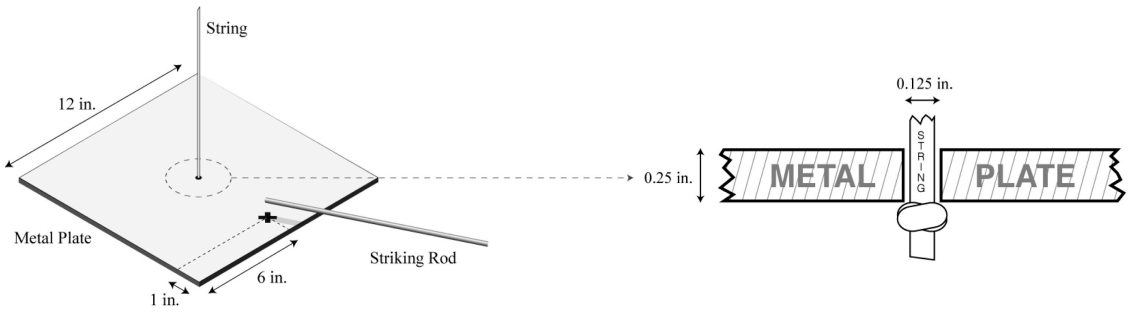

Figure 4.3. Location of the strike on the plate.

4.3 Sound Analysis

We chose to analyze the period of audio immediately after the falloff of spectral transients from the recording. That is because our software does not model the sound emanating from the striking object, nor does it try to capture the complex sounds of object friction. Figure 4.4 demonstrates this audio cropping. In a typical analysis, we cropped the first 0.5 seconds from each sound file and analyzed the subsequent 2 seconds of audio. The files were analyzed using Praat software.

The microphone used for recording was an Electrovoice model number RE20. The microphone was placed in the near field which is contrary to our model which simulates a far-field response. We hoped that this discrepancy would not dramatically alter our results.

4.4 Model Computation

Considerable care was taken to ensure the constraints owing to suspension of the object and location of the strike were the same as those in the physical experiments. When performing the sound-synthesis computations, there are four parameters (corresponding to the experimental variables that are approximately known) that can be adjusted to affect

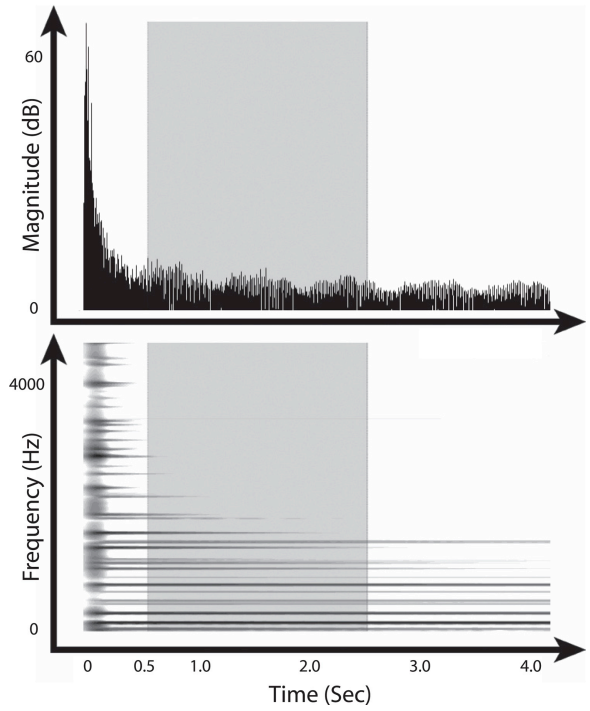

Figure 4.4. Audio amplitude and frequency distribution as a function of time after strike; the time interval chosen for audio evaluation is highlighted.

the resulting frequency spectrum. The first is the materials stiffness. This parameter affects the overall frequency spectrum by scaling the partials along the frequency axis. The second is the material damping parameters, α_1 and α_2. These parameters affect the decay behavior of the oscillations. The other parameters are the location of and the intensity of the object strike. These parameters can be adjusted owing to the uncertainty of their exact values when recording. The strike location affects which modes are activated, and the intensity determines the energy that must be damped and the magnitude of the resulting sound.

We can fine-tune each of these parameters to bring the computed audio into as much agreement with the recordings as possible. (This tuning is only necessary to recover the parameters that are only approximately known during recording. It is a step only necessary for validation of the model and should not be confused with tuning of digital oscillators

that would occur when modeling the object using a bank of second-order resonant filters.) Typically, we map the strike location as accurately as possible and then adjust the material stiffness first to match the first partial in the recording. Then, we adjust the material damping and strike intensity to achieve a similar audio-waveform profile.

The synthetic strike location is chosen to match the actual strike location by measuring the distance of the strike from the edge, and placing a marker to apply force at that point. We take considerable effort in selecting this location, as even slight modification of this point will cause a difference in the frequency spectra in both real and simulated objects. For example, Figure 4.5 shows the effect of striking the square plate at two different locations during an actual recording. The strike location was approximately 3.5 inches apart on the centerline of the square plate. The hatched-line region shows striking the plate 1.4 inches from center, and the solid region shows striking the plate 4.9 inches from center. We can also see in our simulations the effect of these two different strike locations (Figure 4.6).

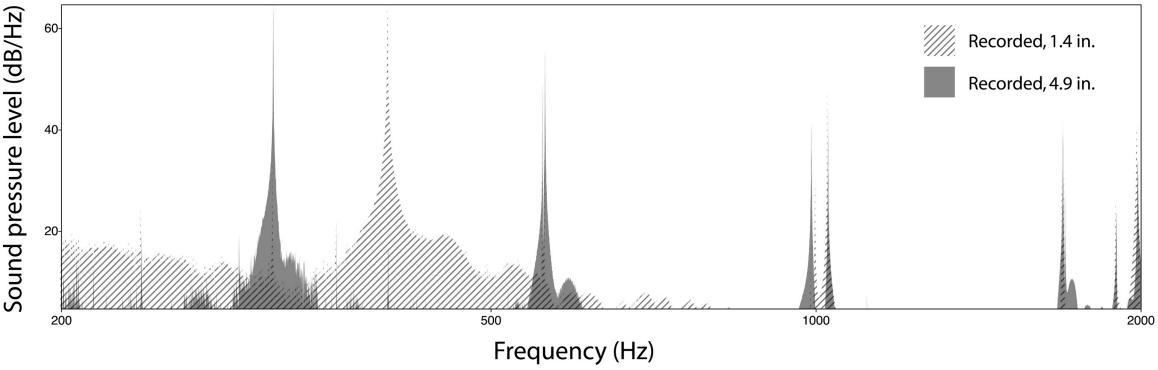

Figure 4.5. Striking a steel plate at two different locations from recording.

Moving the point by an even smaller amount can still change the spectrum as seen in Figure 4.7. The added region shows the effect of moving the outer striking point in toward the center by 0.7 inches (from 4.9 inches to 4.2 inches). Again, the hatched-line region

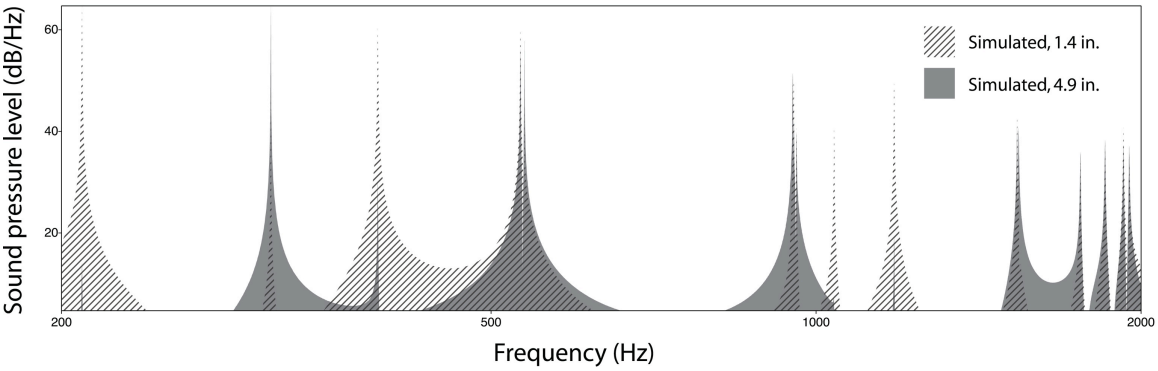

Figure 4.6. Striking a steel plate at two different locations using simulation.

shows striking the plate 1.4 inches from center, and the solid region shows striking the plate 4.9 inches from center.

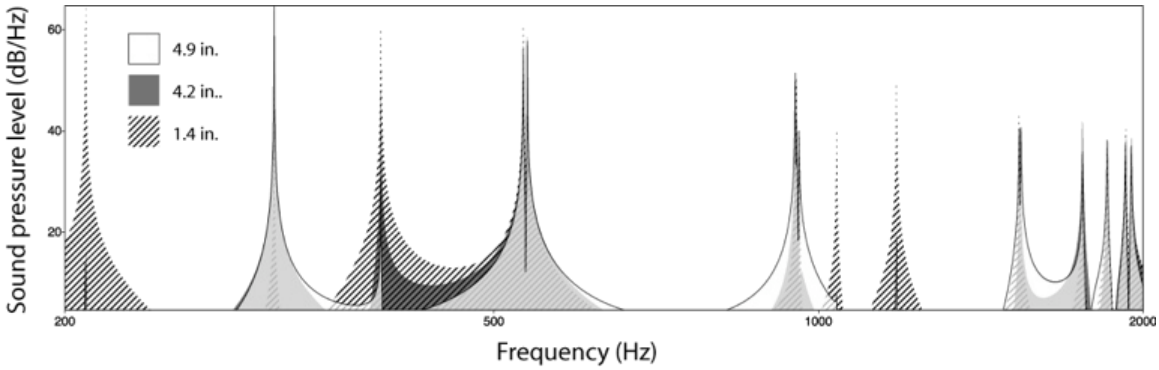

Figure 4.7. Striking a steel plate at three different locations (two very close by) using simulation.

It should be noted that only the magnitudes and phases of the resonances change with different strike points, while the frequency locations remain fixed. From a perceptual perspective, these changes will affect the timbre, but the differences will be small. For example, when plucking a guitar string at two different locations, the sounds may differ spectrally, but they would not affect categorical perception (for example, an acoustic guitar would not be confused with a banjo). In other words, the instrument identity is not conveyed by the attributes of the steady-state partials.

Once the strike location was found, we adjusted the material parameters by moving the first partial of the simulation into alignment with the recording. We then adjusted the damping parameters so that the object resonated for approximately the same amount of time. In this way, the ability of the simulation to match second and higher partials adds to the accuracy of the model.

For the simple shapes, we computed the theoretical natural frequencies using tables provided in Leissa, 1969 [53] and overlapped them on the experimental data and the simulation.

4.5 Evaluation of Spectrum Prediction

Figure 4.8 shows the results of comparing recorded and simulated 1-foot × 1-foot × 0.25-inch square plates made from aluminum and steel. The results show reasonable agreement using a model containing only 500 elements. The largest variation in spectral peaks for the aluminum plate (Figure 4.8 top) was at the second partial frequency (marked with a ×). Here, the computed value (hatched-line region) varied from the recording (solid region) by 6%. For the steel plate (Figure 4.8 bottom), the largest difference was at the third partial frequency (marked with a ×). Here, the model varied from the recording by 2.5%. Theoretical FFFF (for free on all four sides) modal frequencies are shown by solid vertical lines with markers at the top of the plot.

Next, we modeled two complex shapes and compared their spectra with their respective physical models. The first was an aluminum S-shape, and the next was an aluminum G-shape. We then modeled these same shapes using steel material parameters. Figure 4.9 shows these results for two models each having 400 elements. The aluminum S-shape

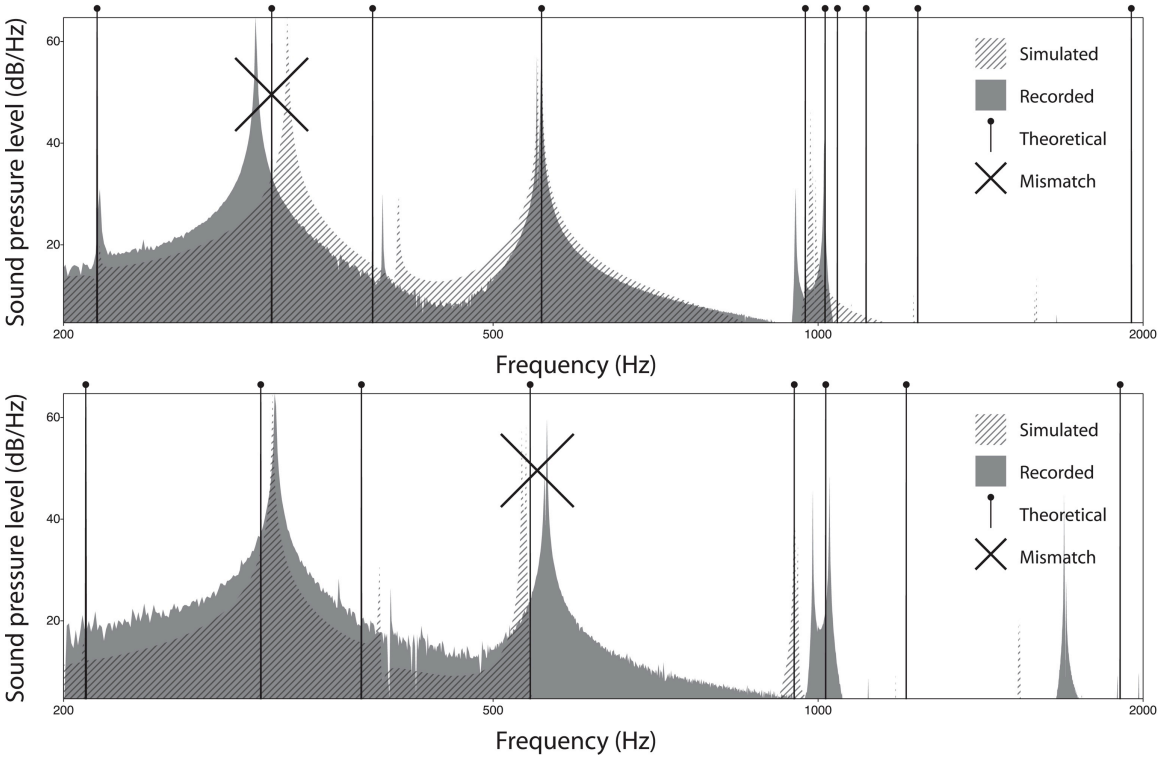

Figure 4.8. Square plate frequency spectrum comparison. Top: Aluminum. Bottom: Steel.

(Figure 4.9 top) matched the recordings first four partials (marked with a ◯), and then began to drift with the upper partials by 5%. The G-shape (Figure 4.9 bottom) matched three of the partials (marked with a ◯) but varied from two others by 7%. The solid regions show the measured values, and the hatched-line regions show the results of the simulation.

The steel S-shape (Figure 4.10 top) matched five partials (marked with a ◯), and then started to drift on the upper partials by 5%. The steel G-shape (Figure 4.10 bottom) only matched two partials (marked with a ◯) before drifting.

Figures 4.11 and 4.12 show the first four mode shapes of the S-shaped and G-shaped plates.

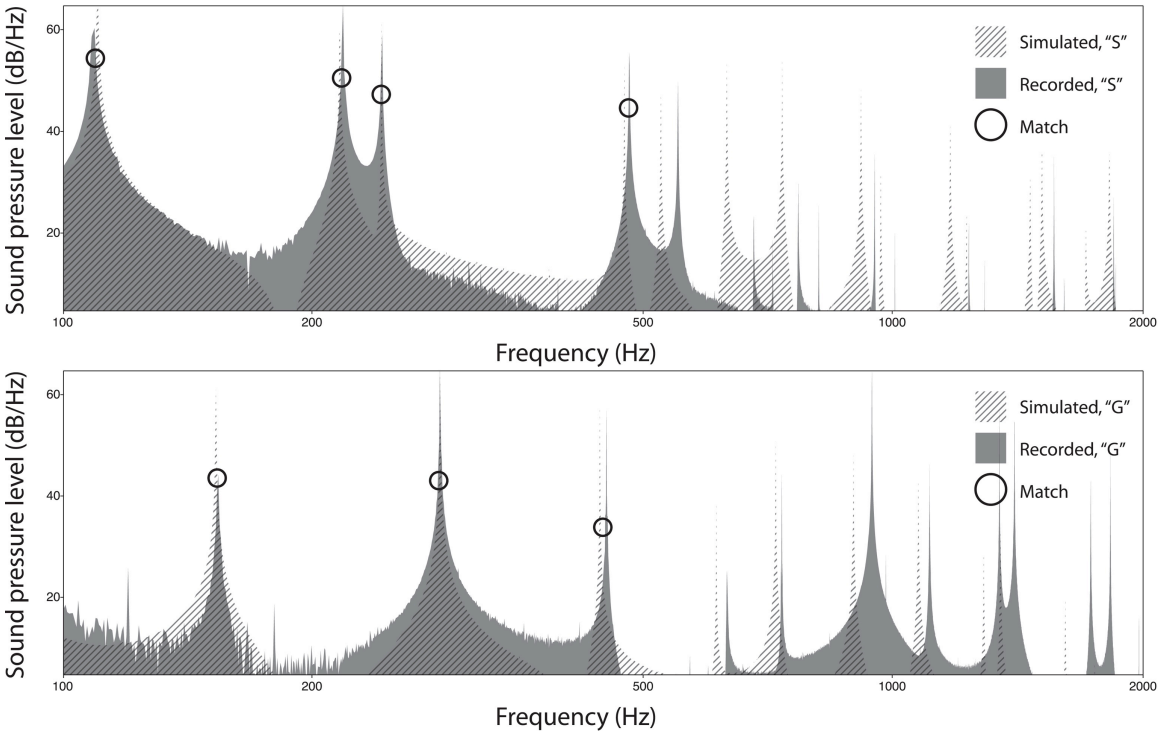

Figure 4.9. Shaped aluminum plates frequency spectrum comparison. Top: S shape. Bottom: G shape.

4.6 Discussion

When evaluating these results, our goal is to match many more of the natural frequencies between the recorded and synthesized sounds, as this is the measure of how well a virtual instrument would represent a real instrument while being played. The rough mapping between the theoretical and recorded audio spectra highlights the need for actual instrument recordings for comparison. These results show that the model matched the predicted lower frequencies exactly and drifted by roughly 7% on the higher frequencies. Some partials were not predicted by our models, and further research is needed to determine what features of the model can be improved to recover these partials. More investigation is also needed on the perceptual implications of these results. The degree to which these spectral variations are perceptible is a complex issue as described by Cook, 2001 [26] and Horner et. al., 2004 [40].

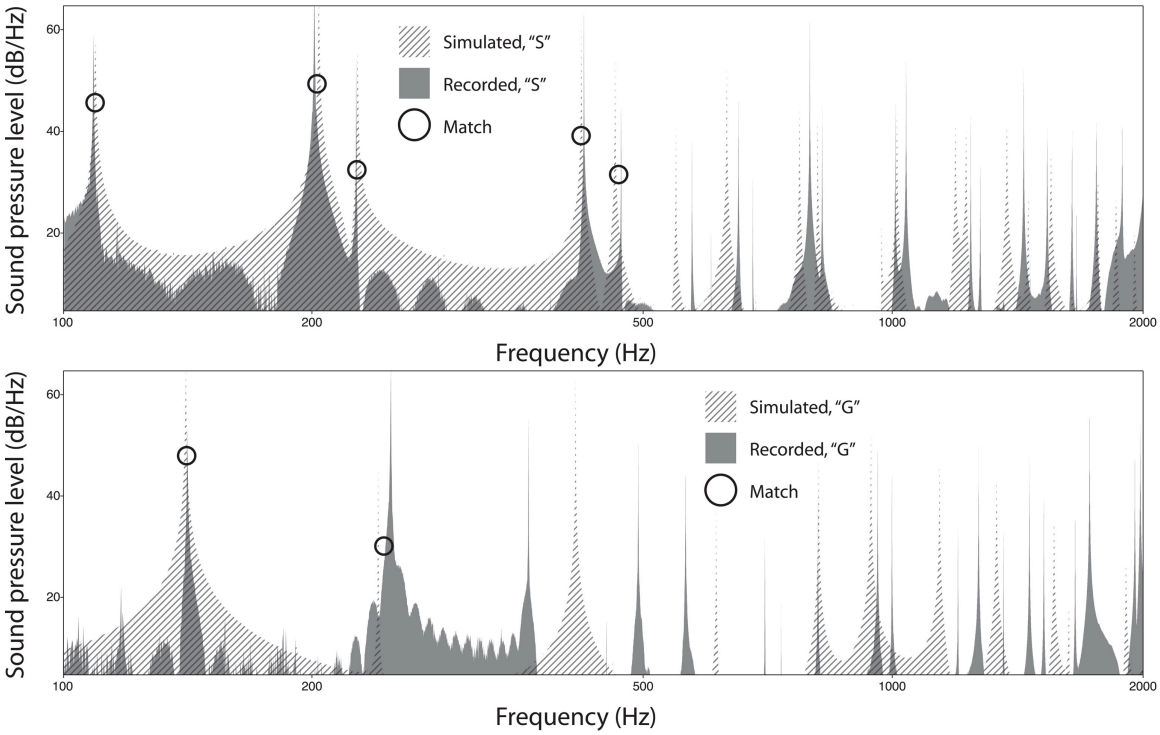

Figure 4.10. Shaped steel plates frequency spectrum comparison.Top: S shape. Bottom: G shape.

When playing the recorded and synthesized audio files to a group of trained and untrained listeners, the only perceptible difference was in the attack period of the strike. Some trained listeners could only tell the difference between the two audio files because the synthesized sound had a very prominent attack, whereas the recording sounded like a more natural strike. Other trained listeners commented that they heard a mechanical quality to the simulations, which might be attributed to the decay model of the resonators. Untrained listeners were unable to distinguish between the two samples. Future research will investigate a more realistic nonlinear decay.

Future experiments will also focus on modeling the transient behavior of the plate strike, as that is the region of time with the most expressive musical content for many percussive instruments. We will also focus on the modeling of the strike object and capturing a more

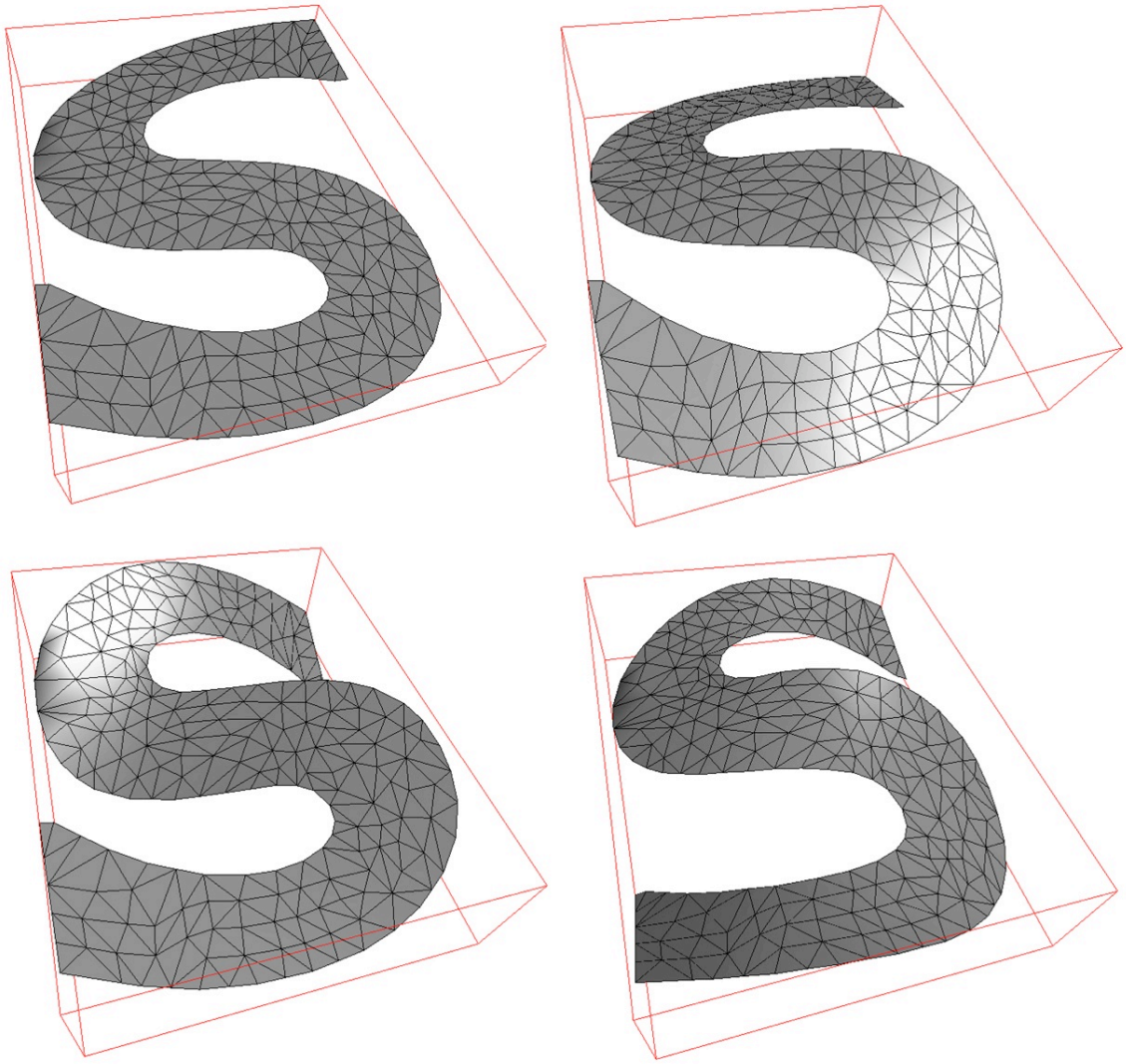

Figure 4.11. First four mode shapes of S-shaped plate.

complicated applied force profile using the techniques presented in Chapter 2.2. Accurately modeling this interaction will be crucial in providing a realistic percussive instrument simulation.

We use this chapter as a framework that validates the finite element models used in this thesis. The next two chapters present our original contributions that build on the finite element and modal analysis methods for sound synthesis. These methods extend

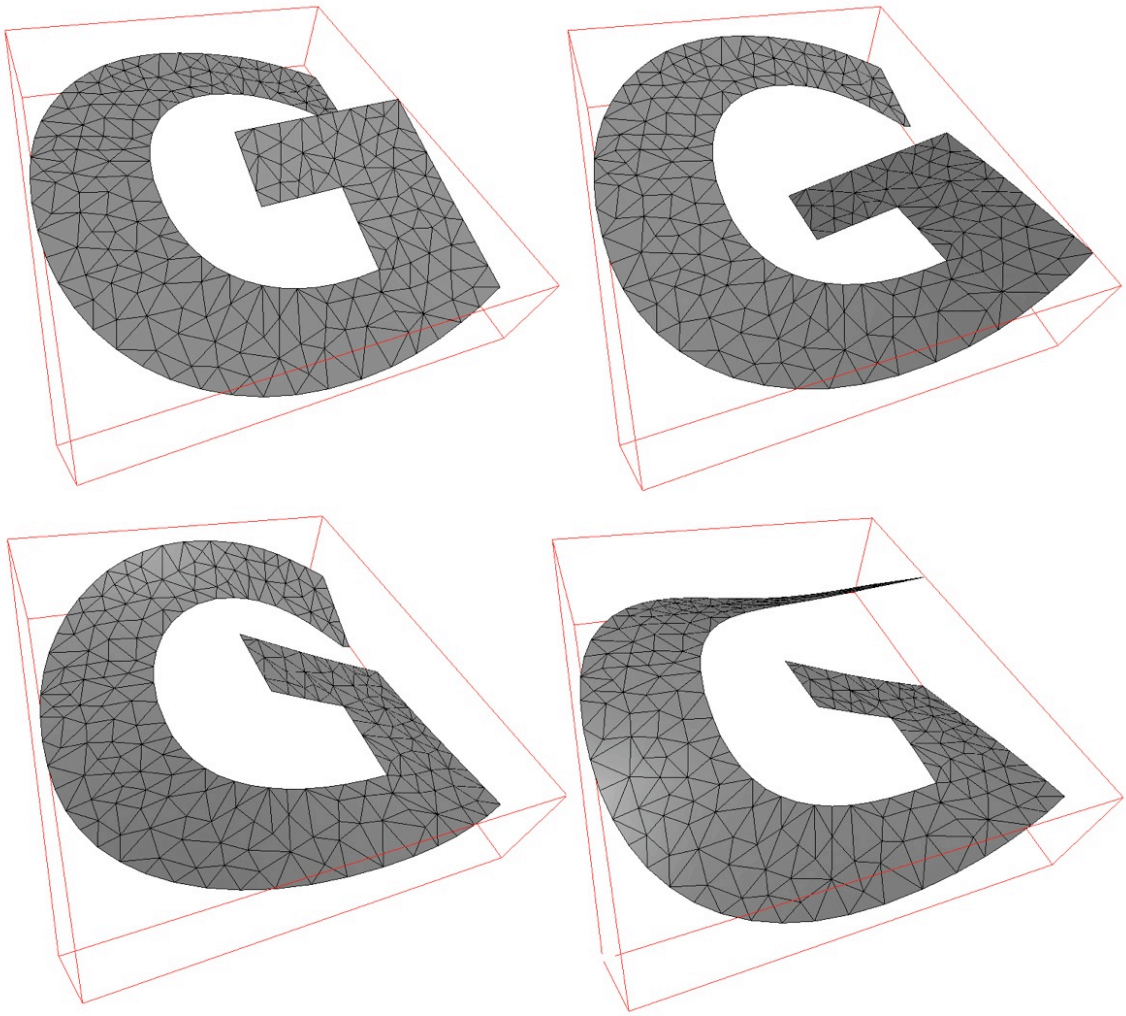

Figure 4.12. First four mode shapes of G-shaped plate.

the previous sound synthesis methods to capture the behavior of objects as their shape is changing. Using these techniques one can rapidly audition the timbre of an object inside of a sound synthesis environment.

Chapter 5

Geometric Modifications for Axisymmetric Resonance Models

In this chapter, we describe how to quickly compute modes of axisymmetric objects in order to synthesize sounds of new instrument shapes quickly enough for interactive instrument design. We use the bell as a canonical axisymmetric instrument for exploration of this technique.

The bell has been a fascinating topic of mathematical and artistic research for decades. There have been many studies on the vibration of bells[1]. Some studies seek to classify the different vibrational modes of bells; others try to understand the importance of curvature on the overall bell tone; while still others seek to design a bell with a given harmonic series through shape optimization. We use this literature to validate our models for axisymmetric object vibration.

We separate our analysis into thick and thin axisymmetric bells to demonstrate two methods for performing fast eigendecomposition. The first technique described in Sec-

[1]See references [81], [67], [69], [74], [52], [73], [68], [60], [61], [65].

tion 5.1 separates out the rotational symmetry after finite element discretization. The second technique, described in Section 5.2, separates out the radial symmetry as part of the finite element formulation.

5.1 Thin Axisymmetric Objects

Many instruments, such as smooth bells, are rotationally symmetric. If we choose a finite element discretization that preserves this symmetry, we can perform a numerical separation of variables to expedite eigendecomposition.

We first change to a cylindrical coordinate system, in which the system matrices have a block circulant form. These block 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 [29]. We can then compute a small modal decomposition for each of these independent sub-blocks instead of directly computing one large modal decomposition for the entire system. This is physically equivalent to examing the object one azimuthal wavenumber at a time.

By using symmetry in the modal analysis of these objects, each geometry can be analyzed quickly enough for interactive use. For example, for a bell model with over 2000 degrees-of-freedom (DOFs), we can compute the complete modal decomposition in just under five seconds. Using an ordinary eigenvalue solver, the same computation would take over 16 minutes. [2]

[2]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.

5.1.1 Method

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

$$M^e = \int_\Omega N^T \rho N d\Omega \tag{5.1}$$

$$K^e = \int_\Omega B^T D B d\Omega \tag{5.2}$$

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 symmetric gradient operator on the interpolation functions.

Assembling these matrices into a representation for the entire system yields

$$M\ddot{u} + Ku = F(t) \tag{5.3}$$

where M is the system 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:

$$Q_R = I \otimes Q \tag{5.4}$$

where

$$Q = \begin{pmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.5}$$

In this way:

$$A = K_{11}, \quad B = K_{12} Q_R, \quad B^T = Q_R^T K_{21} \tag{5.6}$$

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:

$$K = \begin{pmatrix} A & B & 0 & ... & ... & B^T \\ B^T & A & B & 0 & ... & 0 \\ 0 & B^T & A & B & 0 & ... \\ ... & ... & ... & ... & ... & ... \\ ... & ... & 0 & B^T & A & B \\ B & 0 & ... & 0 & B^T & A \end{pmatrix} \tag{5.7}$$

This form can also be expressed compactly as:

$$K = I \otimes A + P \otimes B + P^T \otimes B^T \tag{5.8}$$

Where P is a cyclic permutation matrix and $\otimes$ is the Kronecker product such that [29]:

$$I \otimes A = \begin{pmatrix} A & 0 & ... & ... & ... & ... \\ 0 & A & 0 & ... & ... & ... \\ ... & 0 & A & 0 & ... & ... \\ ... & ... & ... & ... & ... & ... \\ ... & ... & ... & 0 & A & 0 \\ ... & ... & ... & ... & 0 & A \end{pmatrix} \tag{5.9}$$

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

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

Where L is a diagonal eigenvalue matrix associated with Z:

$$\text{Ł} = \begin{pmatrix} 1 & ... & 0 & ... & ... & 0 \\ 0 & \omega & 0 & 0 & ... & 0 \\ 0 & 0 & \omega^2 & 0 & 0 & \\ ... & ... & ... & ... & ... & ... \\ ... & ... & 0 & 0 & \omega^{n-1} & 0 \\ 0 & 0 & ... & 0 & 0 & \omega^n \end{pmatrix} \tag{5.10}$$

Now computing the eigensystem for the original system can be done for each radial wave number on the matrix pencil.

$$K = A + L(k,k)B + L(k,k)^* B^* \tag{5.11}$$

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 reduced system are the same as the eigenvalues of the original system.The eigenvectors are recovered by the projection:

$$X_{original} = (X_{reduced})(Z \otimes I) \tag{5.12}$$

5.1.2 Results

We can employ this reduction technique by parametrically controlling the height, and radii of four control points of an interpolating B-Spline curve as shown in Figure 5.1.

This curve is then swept 360-degrees in 40 angular divisions about the z-axis, creating the shell geometry shown in Figure 5.2. We use the thin shell model described in Kwon and Bang [49]. This model has six DOFs at each vertex representing three translations and three rotations.

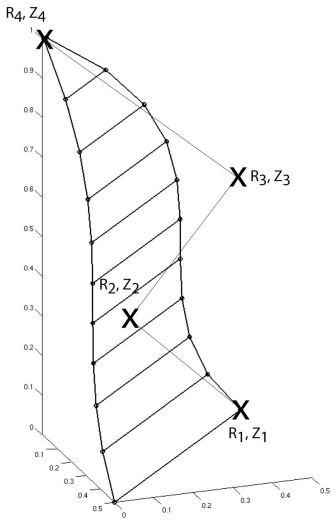

Figure 5.1. Parametric curve.

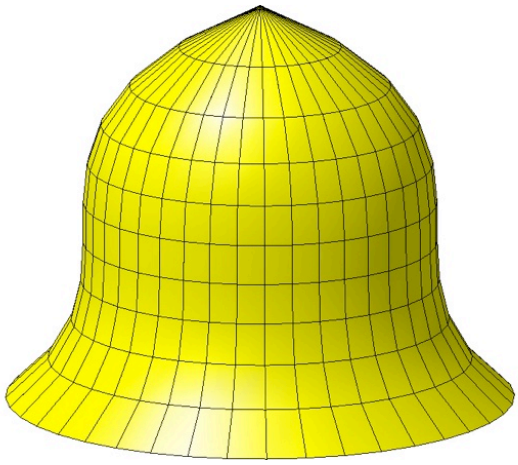

Figure 5.2. Undeformed bell created from swept curve.

Using this radial analysis technique, the original stiffness matrix of the system can be represented as 40 sub-blocks, each of size 66×66 (40 angular divisions, 11 lateral divisions, six DOFS per node) instead of one large $2{,}640 \times 2{,}640$ 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 on a 1.67 GHz PowerPC G4. By exploiting the symmetries of the object, we can find exactly the same resonant information of an object in a fraction of the time it would take using traditional modal analysis methods.

Moreover, because we factor out the object symmetries, each sub-block is also free from repeated eigenvalues, which is favorable when tracking an invariant subspace of these objects (as will be described in Chapter 6).

5.2 Thick Axisymmetric Objects

In this section we examine axisymmetric models that have thick cross-sections unlike the thin shell models examined previously. We use a canonical bell shape as an example and describe an optimization that would allow for these models to be modified interactivley. We compare the accuracy, size, and speed of evaluating a full geometric discretization versus a model that exploits axisymmetry. The result is an efficient representation that can be used for interactive bell design.

We could use the same radial decomposition technique described in the previous section for thick axisymmetric objects, but instead in this section we discuss another optimization which exploits object symmetry in the finite element discretization.

We can extend also our notion of axisymmetric objects to include objects with variable

thickness cross-sections as well. This is in contrast to our thin shell models previously presented which assumes a constant thin cross-section through the shell element.

We begin by analyzing a solid model of a bell and then propose an optimization to allow for faster analysis of the modes of vibration. This optimization makes it possible to rapidly analyze the geometry as well as maintain high accuracy in the model.

5.2.1 Methods

Before we examine the axisymmetric optimization, we will examine the prediction capabilities when using a full 3D tetrahedral model.

We generated a 3D model of a Taylor Carillon 6/5 European Profile Bell (http://www.taylorbells.co.uk/) by revolving a curve describing the bell profile (in DXF format) 360-degrees (courtesy Neil Mac Lauchlan). The resulting solid region was then meshed with unstructured tetrahedra and contained $2,000$ elements (Figure 5.3).

Using the solid tetrahedral finite element formulation as described in Bhatti, 2006 [17], and performing a modal decomposition as described in Chapter 2.1 we compare our predictions against the measurements taken by Perrin et. al., 1983 [69].

5.2.2 Results

Figure 5.4 shows, reasonable agreement between the study presented in Perrin et. al., 1983 [69] and our finite element model. Figure 5.4 also shows that our finite element solutions converge to the experimental values (and assumed exact solutions) from above, which is expected. Linear tetrahedral conforming elements require C^0 continuity across element boundaries and C^1 continuity inside element boundaries. These conforming elements are

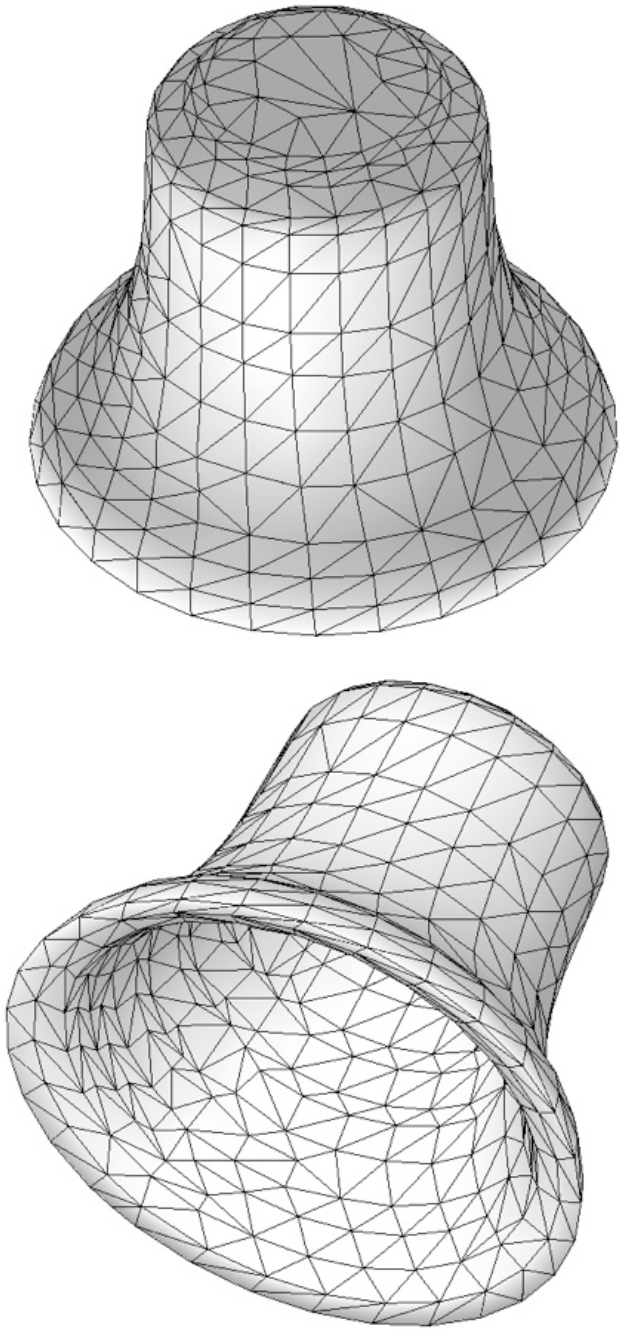

Figure 5.3. Solid model of a D5 Taylor Bell: 2,000 elements.

generally known to add stiffness to the model, which means that they will generally over-predict the resonant frequencies of an actual object. It is generally known that refining the element sizes will control the added stiffness in such a model such as described in Section 2.1.3.

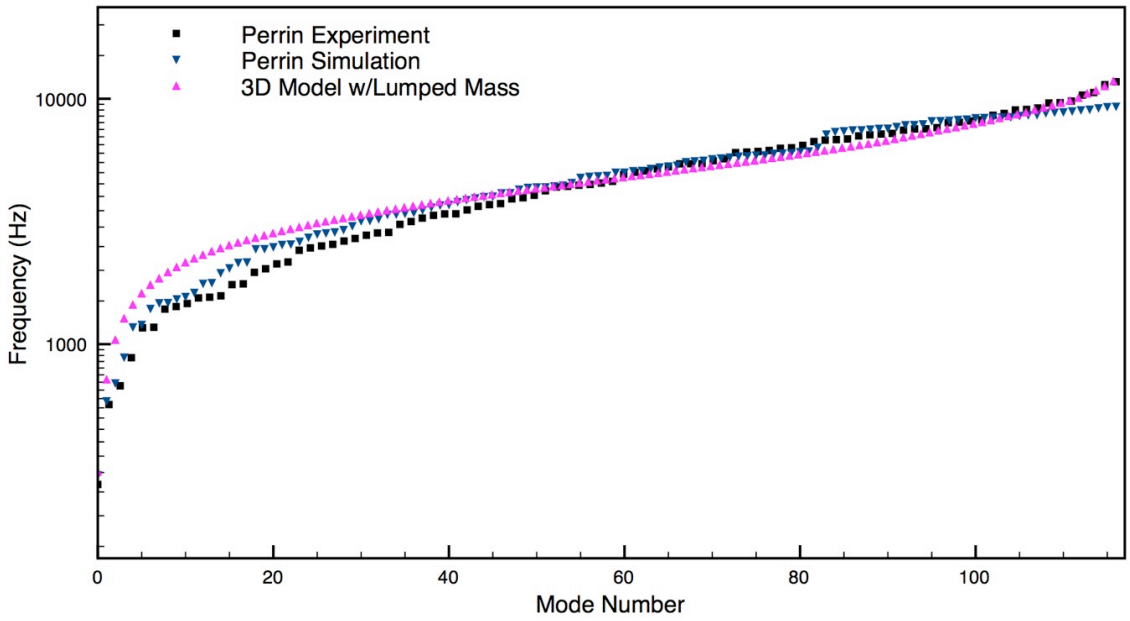

Figure 5.4. Comparison of our model using a lumped mass formulation with with measured and simulated resonant frequencies.

In the next sections we compare the results of our model to the experiment and simulation presented in Perrin et. al., 1983 [69].

5.2.3 Probable Causes for Error

When measuring the accuracy of a finite element model in predicting measured or theoretical results, there are several possible sources of error that must be explored. The following sections demonstrate how one can check a model for these sources of error, and correct them to obtain accurate predictions.

Error in Finite Element Code

A straightforward source of error could be in the implementation of the finite element equations. To test that our program was correct, we compared the mass and stiffness matrices generated by our program against the ones generated by the MATLAB implementation provided on the CD ROM in Bhatti, 2006 [17]. This test eliminated coding mistakes as the likely source of observed errors.

Error in Material Model

The material parameters of the object being modeled are known only approximately. That is because in many instances, the exact mixture of alloys in a material and therefore the resulting elastic modulus is not known. In many cases testing to discover the actual elastic modulus is performed on a sampling from a batch of material produced. For the bell we are modeling, which was fabricated decades ago, obtaining such a sample is not possible. However, by using the published parameters, there is no need for tuning the material properties as was performed in Chapter 4.

Error in Finite Element Discretization

There have been many studies on the effect of spatial and temporal discretization on the quality of finite element approximations [39], [15], [16], [55]. These studies find that the accuracy of a finite element simulation depends on the finite element discretization of the domain. Specifically the discretization should allow the deformation to capture propagation of a traveling wave with a given wavelength.

The wavelength of waves traveling in a medium can be calculated as:

$$\lambda = c/f \tag{5.13}$$

where c is the speed of sound in the medium and f is the frequency of interest. Because only those frequencies that can be captured by the discretization are represented, spatial discretization acts as a low-pass filter [42].

We would like to be able to resolve deformations of the object corresponding to frequencies up to the ones measured experimentally. We can determine c from known material properties by:

$$c = \sqrt{E/\rho} \tag{5.14}$$

where E is the Young's modulus and ρ is the density of the material. Using the material values for Brass Alloy published in Perrin et. al., 1983 [69], E = 103 $\times$ 10^9 Pa and ρ = 8.85 $\times$ 10^3 kg/m^3, which gives c = 3411.5 m/s.

Therefore, to resolve frequencies up to 9.3 kHz we need to be able to capture a wavelength of λ = 0.367 meters or 36.7 centimeters. In order to have 10 elements per wavelength (which is commonly used in mechanical engineering analysis), we would need the average element length to be around 3.67 centimeters. This means that for the bell, with diameter of 70.2 centimeters and a height of 56.6 centimeters, we can approximate the cylindrical surface area to be 12,483 centimeters^2. This means we would need roughly 3,401 elements to fully resolve the highest measured frequency. Despite falling short of this requirement, our model matches the simulations performed in Perrin et. al. [69] quite well.

Error in Mass Matrix Formulation

Studies by Tong et al. [77] have found that the mass matrix formulation can affect the rates of convergence of the mode shapes and frequencies in finite element models. Specifically they determined that the interpolation functions q must satisfy the following relation to avoid loss of convergence when using a lumped mass formulation.

$$\bar{q}+1-\bar{p} \quad \geq \quad 2(\hat{q}+1-\hat{p}) \tag{5.15}$$

where p is the highest partial derivative in the functional operating in the weak form of the model. Returning to the equations of linear elasticity described in Kwon and Bang [49] we see that for these systems $p=1$. Using the interpolation functions described for solid 4-node linear tetrahedral elements, we have $q=1$. Plugging these values into Equation 5.15 we see that although these interpolation functions satisfy approximation completeness, a lumped mass formulation will suffer loss of convergence using only linear elements. We can confirm that a consistent mass formulation will perform better by comparing the resonant frequencies again this time using a consistent mass formulation. The results in Figure 5.5 show improved matching between the 2,000 element model and the data presented in Perrin et. al., 1983 [69].

5.2.4 An Optimization

Creating a discretization for this entire geometry is inefficient in storage and the time it takes to compute the decomposition. If we instead exploit geometric symmetry, we can reduce the overall size of the system and time to compute the modes of interest. This is accomplished by changing the discretization of the finite element equations of motion into axisymmetric form as described in Bhatti, 2006 [17]. This axisymmetric formulation differs

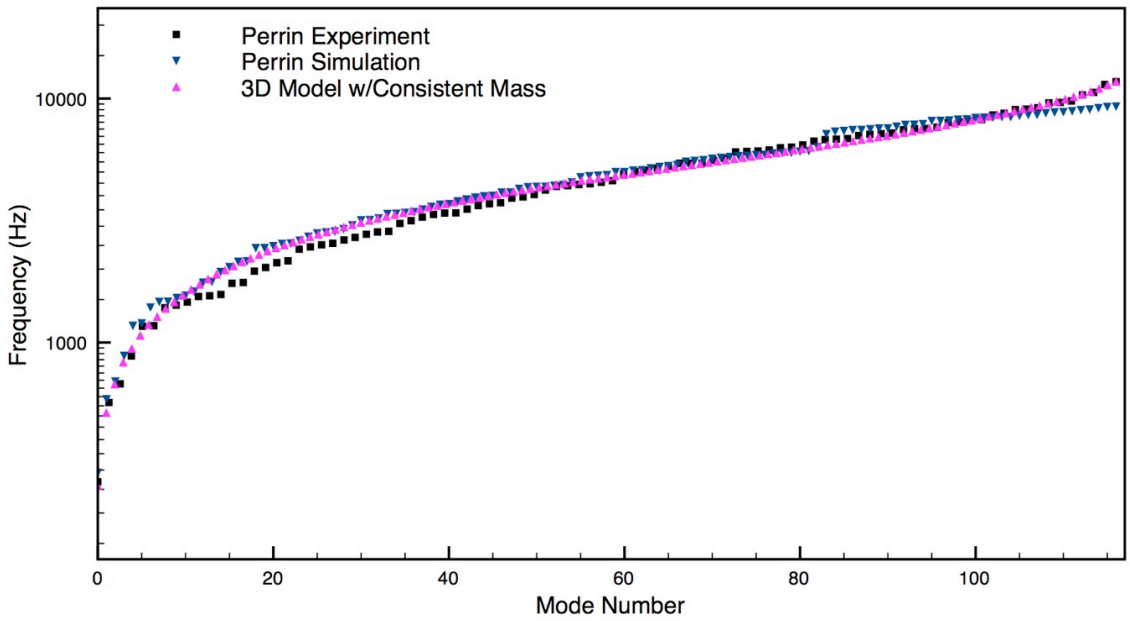

Figure 5.5. Comparison of our model using a consistent mass formulation with with measured and simulated resonant frequencies.

from the symmetric decomposition as described in Chapter 5.1 in that is exploits symmetry before the equations are transformed into an algebraic system. That is, by using a finite element representation that discretizes the partial differential equations in a manner that preserves and parameterizes the object's symmetry, one can perform radial analysis one angular wave number at a time. In this way, once can solve several smaller problems and still recreate the original frequency spectrum.

As described in Bhatti, 2006 [17] for a homogeneous, isotropic axisymmetric elastic body, the displacements (u,v,w) in cylindrical polar coordinates can be represented as

$$u = u(r,z,t)\cos(m\theta+\alpha) \tag{5.16}$$

$$v = v(r,z,t)\sin(m\theta+\alpha) \tag{5.17}$$

$$w = u(r,z,t)\cos(m\theta+\alpha) \tag{5.18}$$

where z is the axis of symmetry and r points in the radial direction. The number m is the

circumferential wavenumber; it is thus the number of nodal meridians for a given mode shape [60]. For example, Figure 5.6 shows the nodal meridians (dashed lines) for $m = 2$, ,3, 4, and 5.

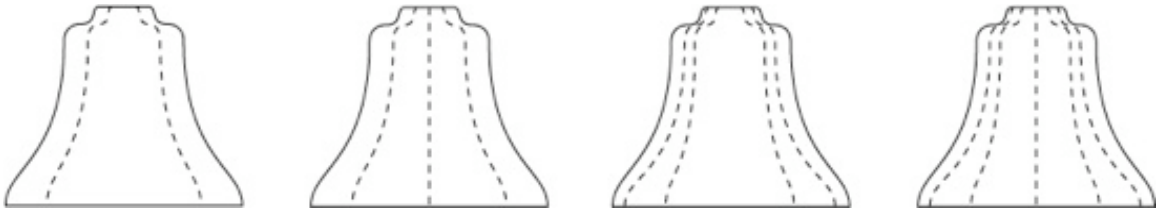

Figure 5.6. Vibrational modes of a bell illustrating the different circumferential wavenumbers. Dashed lines show nodal medians for $m = 2$, ,3, 4, and 5.

For investigation using the optimized finite element formulation, we used the bell cross-section as defined in Perrin et. al., 1983 [69]. A list of points and connectivity for a mesh containing four-node quadrilateral and serendipity (elements with added nodes to facilitate mesh generation) elements was provided by Robert Perrin. The connectivity information was then edited to generate models containing only four-node quadrilateral or only three-node triangular elements.

The triangular element mesh was created using MATLAB's Delaunay triangulation from the list of points provided. The quadrilateral mesh was generated by hand-editing the provided connectivity data to break the serendipity elements into four-node quadrilateral elements. In this way, the meshes were not identical to those in the study, but they represent approximately the same 2D domain. The resulting meshes are shown in Figure 5.7.

Using these models, we compute the modal parameters for $m = 1$, 2, 3, and 4. Figures 5.8 and 5.9 show the resonant frequency predictions given by the model used by Perrin et. al. [69] and our axisymmetric solid model. These results show that even for a modest number of

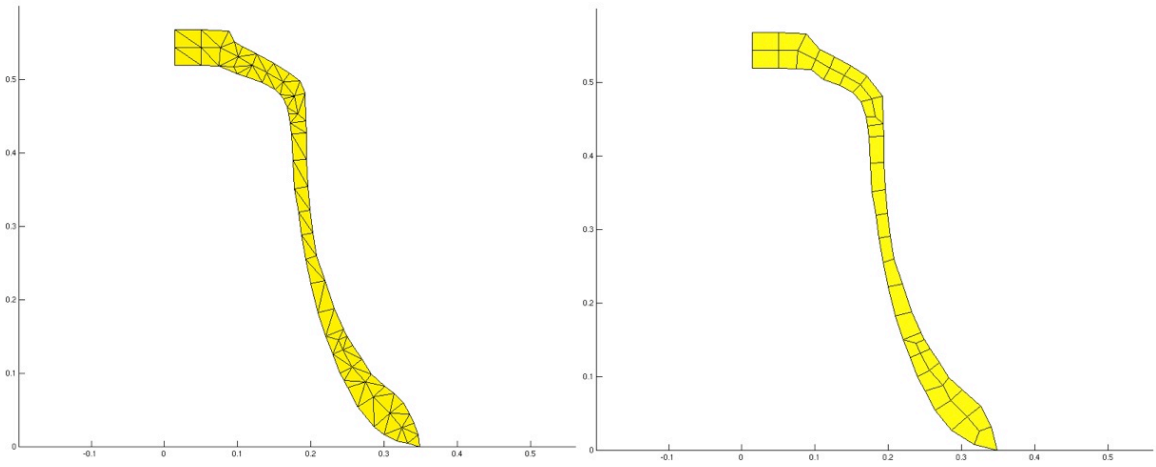

Figure 5.7. Axisymmetric model of a D5 Taylor Bell. Left: Triangular mesh. Right: Quadrilateral mesh.

elements, 92 for the triangle mesh and 40 for the quadrilateral mesh, we still achieve high accuracy in the predicted resonant frequencies.

Both Figure 5.8 and Figure 5.9 show that our axisymmetric finite element models approximately match the simulation performed by Perrin et. al. [69] from above. Figure 5.9 shows that the axisymmetric quadrilateral model differs from Perrin et. al. more prominently than the axisymmetric triangular model. This is somewhat expected as the quadrilateral model uses only one element throughout the profile thickness which adds artificial stiffness to the model. Despite the approximate matching to the simulation performed in Perrin et. al. [69], we are more interested in matching the measured spectral profile.

Combining the predicted frequencies and comparing against the experimental data in Perrin et. al. [69], we see that both of these models perform just as well as the full 2,000 element solid tetrahedral representation (Figure 5.10).

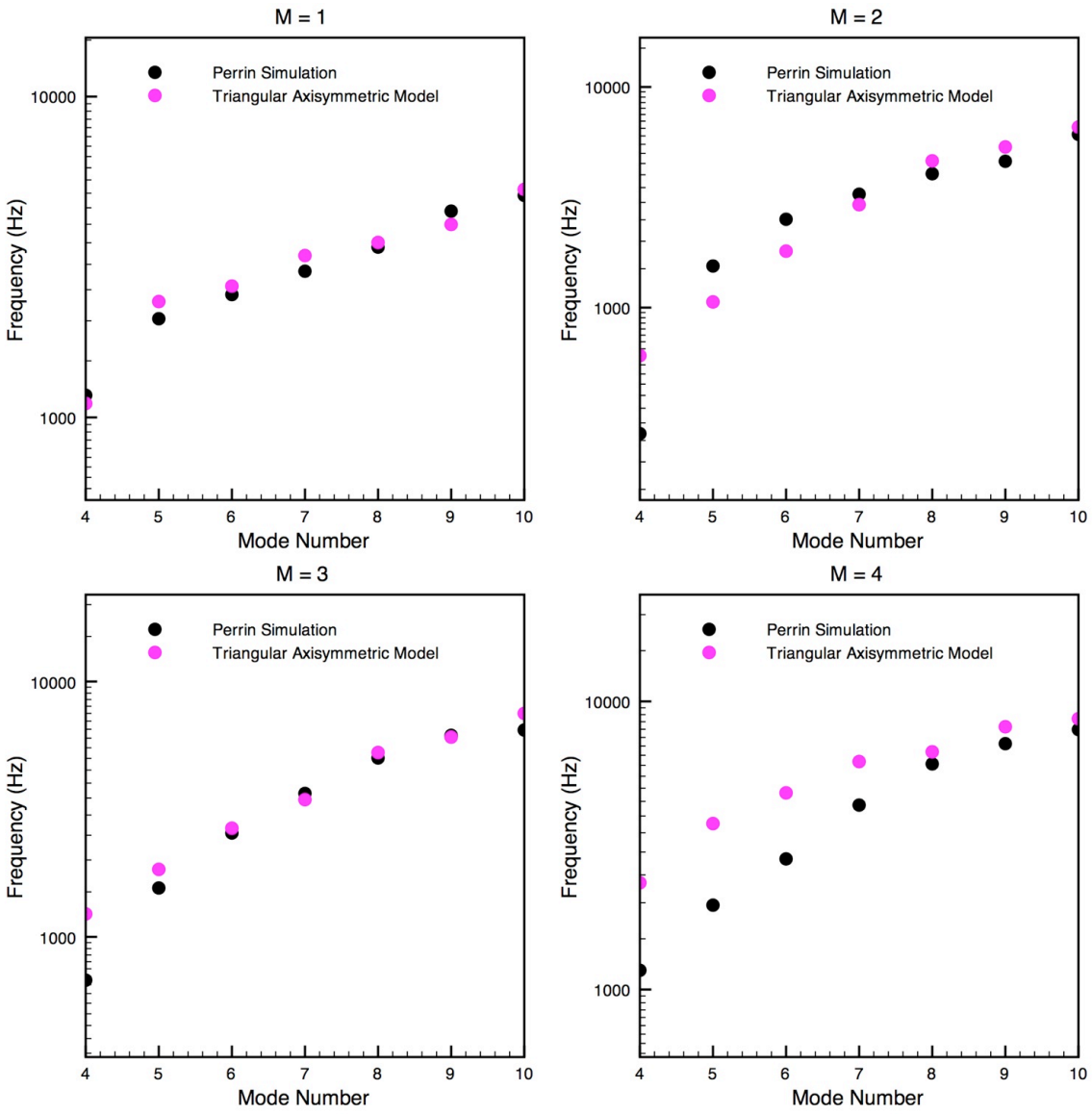

Figure 5.8. Comparison at four different radial slices: Triangular mesh.

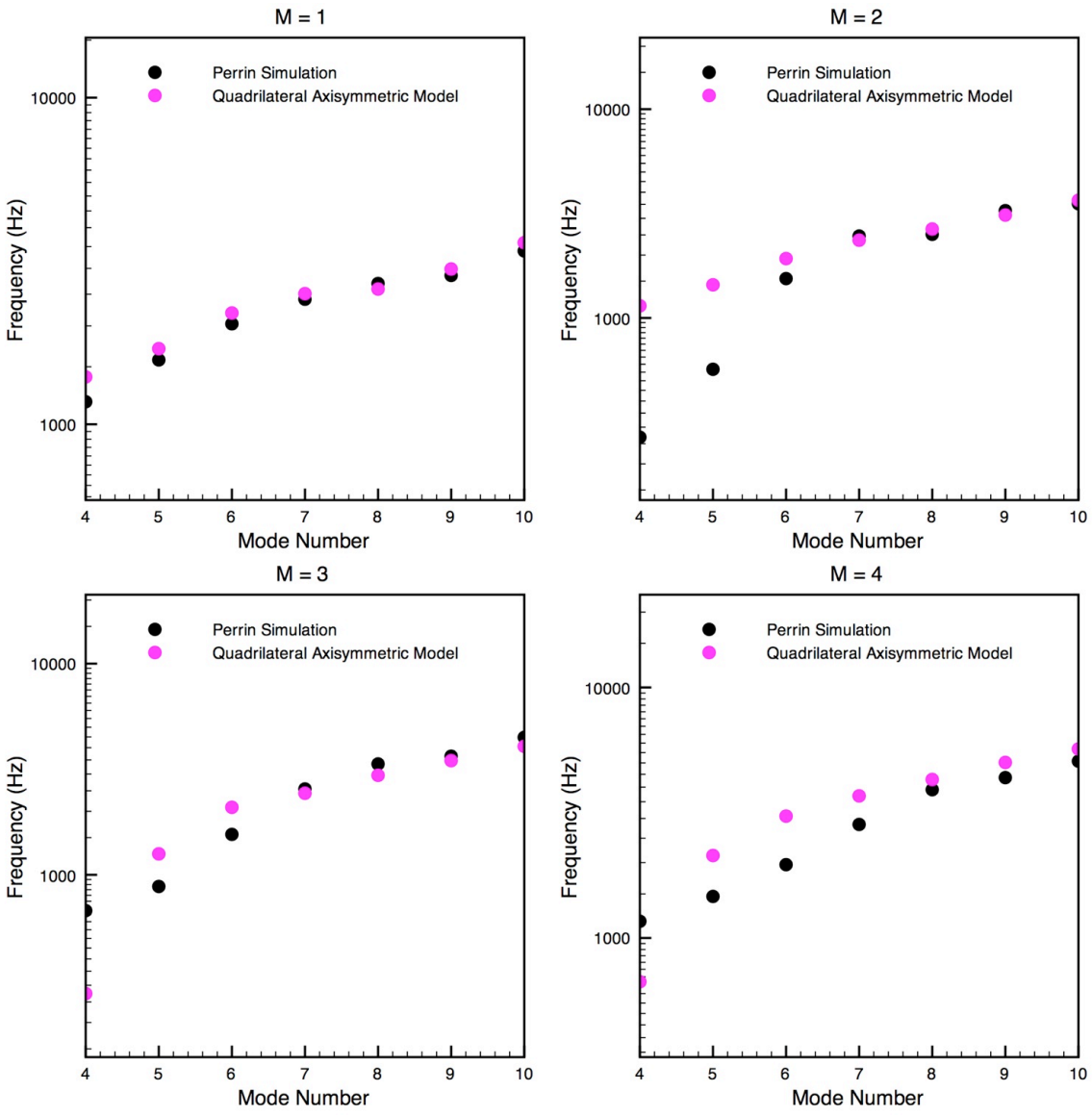

Figure 5.9. Comparison at four different radial slices: Quadrilateral mesh.

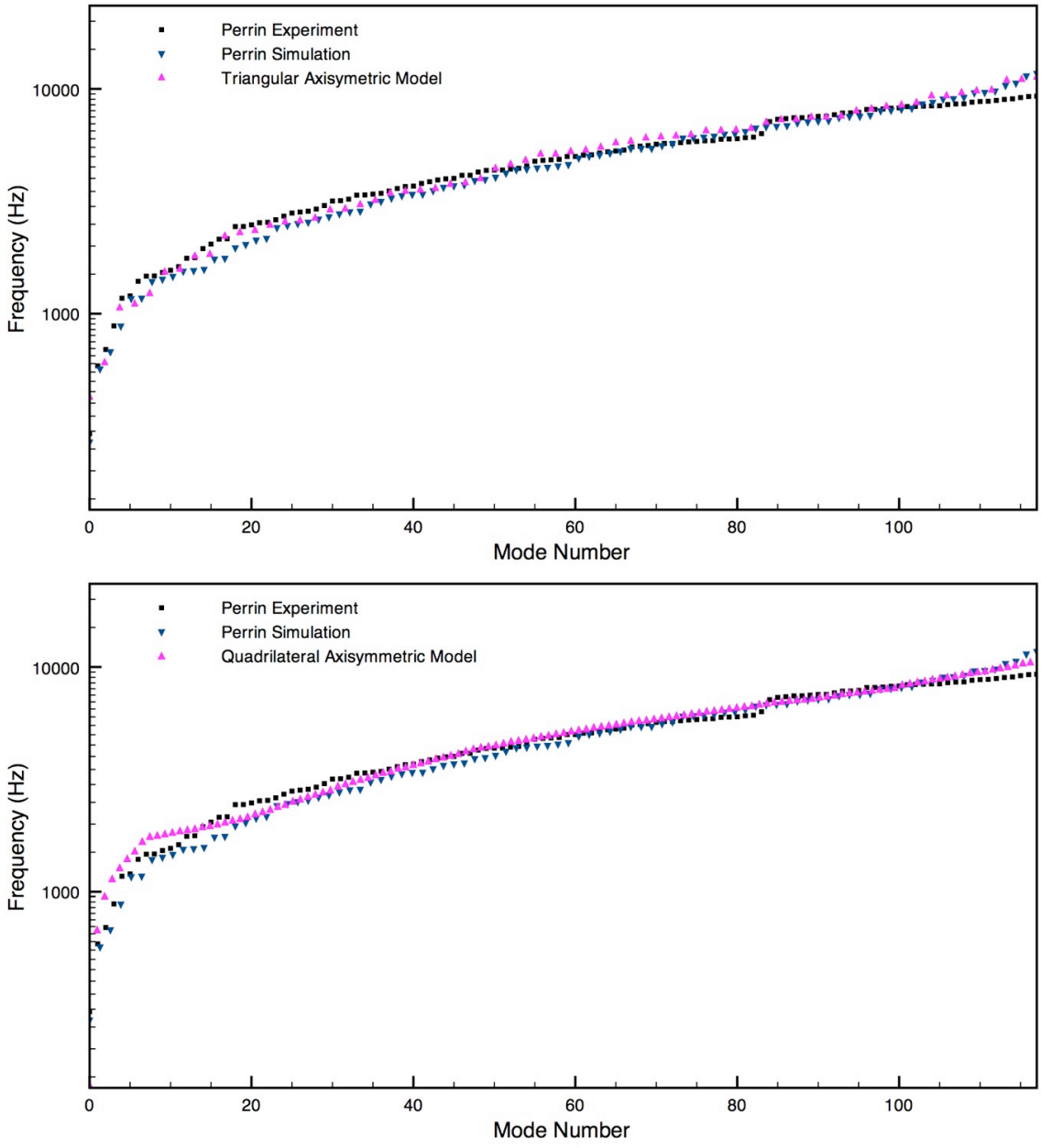

Figure 5.10. Comparison with measured values. Top: Axisymmetric triangular model. Bottom: Axisymmetric quadrilateral model.

5.2.5 Conclusions

Similar to the method presented in Section 5.1, analyzing the geometry for each wavenumber solves a much smaller problem while still retaining the original eigenvalues of the system. In this way, we do not avoid the $O(m^3)$ operations, we just use $m \ll n$, where n would be the number of degrees of freedom in the entire solid model.

This optimization now allows us to investigate modifying geometry with thick profiles and rapidly compute approximations of the new frequency spectrum. In this way, one can change a curve defining an object cross-section or the thickness at various locations and still quickly compute the decomposition. For large changes, a re-meshing algorithm would need to be implemented to avoid element distortion. Future work will examine methods for modifying the geometry while maintaining high quality elements.

In the next chapter we examine another way to avoid a full eigendcomposition as changes are made. This technique also provides methods for transferring information from one mesh to another. In this way, even in the worst case, when remeshing is needed, one can still rapidly approximate the new spectrum of an object.

Chapter 6

Geometric Modifications to General Parametric Resonance Models

6.1 Introduction

Using modal synthesis, the vibration of an instrument can be computed efficiently. To use modal synthesis, however, one must first compute a full or partial eigenvalue decomposition of the system matrices. This eigenvalue problem is relatively expensive, but one only needs to compute the decomposition once for a given instrument.

Eigenvalues and eigenvectors depend strongly on the shape of the instrument, so to design new instrument shapes with standard modal analysis, one would need to recompute the modes for each new design – a prohibitively expensive step for an interactive tool.

The goal of the work we describe here is to design instruments in real time. That is, we would like to know precisely how changing the shape or the material of an instrument will change its timbre.

This chapter describes one method that can be used to predict how the eigenvalues

and eigenvectors will move when the geometry changes by a small amount. We will also show that by concatenating these estimates, one can track even large geometric changes to an object. The method exploits properties of parameter-dependent linear systems by tracking an invariant subspace as modifications are made. Using this method, one can avoid recomputing the spectrum while still providing an accurate representation of the timbre of an object. The results show very high accuracy for moderate changes. Moreover, our algorithm runs in a modest linear time for standard finite element discretizations.

6.1.1 Perception

The methods presented here are strictly concerned with how accurately our system can predict the resonant frequencies from one shape to another. Accuracy is measured with respect to a full eigendecomposition using the method presented in Chapter 2.1. Bear in mind that this notion of accuracy is different than accuracy as measured in Chapter 4. That is, we are interested in the spectrum in the mathematical sense, not the final perceived tone. Issues involving the perceptual differences in timbre given a change in placement or amplitude of a dominant frequency are not addressed in this dissertation. In other words, we are not trying to match a recorded spectrum, only the spectrum as predicted by a full eigendecomposition. Recall that this also means that we not predicting the amplitudes of the dominant frequencies during vibration as this would depend on forces applied during synthesis.

6.1.2 Related Work

It is worth comparing our proposed method with another model reduction technique described in Barbič and James, 2005 [13]. In their paper, they describe a method for

sampling the deformation space of an object to pre-compute typical force vectors for a given user interaction. Using this sampling, they project their full system onto the sampled subspace to achieve fast and plausible interaction results at run-time. This sampling of deformations and projection onto previous solutions is similar to the method we propose; however, we will be sampling the deformations of an object incrementally at run-time, not as a pre-computation. Our method also supports the mapping of deformation variables from one geometry to the next, even in the case when the mesh is changing. We therefore believe that by using our techniques, one can map the polynomial coefficients computed in Barbič and James, 2005 to support interactive change of nonlinear modal models as well the linear models presented here.

6.1.3 Model reduction

Recall that the eigenvalue problem that we want to solve is:

$$Kx = \lambda Mx \tag{6.1}$$

where K (the stiffness matrix) is symmetric and M (the mass matrix) is symmetric and positive-definite, and x is the vector of nodal displacements of the mode with natural frequency $\lambda = \omega^2$.

One means of formulating approximate solutions to this system for freely vibrating discrete systems is via the Rayleigh quotient:

$$\lambda_R = \frac{\hat{x}^T K \hat{x}}{\hat{x}^T M \hat{x}} \tag{6.2}$$

where $\hat{x}$ is an approximation to x [64]. The relative accuracy of methods based upon this formulation results from the fact that eigenvalues λ are stationary with respect to pertur-

bations in the elements of K and M (the eigenvectors x, like the eigenvalues, are functions of K and M – and therefore are not subject to independent perturbations).

Thus, if a transformation for the n physical node displacements, $\hat{x}$, into fewer ($m < n$) generalized coordinates is available, say

$$\begin{array}{ccccc} \hat{x} & = & V & & y \\ n \times 1 & & n \times m & & m \times 1 \end{array} \tag{6.3}$$

then the corresponding Rayleigh quotient becomes

$$\lambda_R = \frac{y^T V^T K V y}{y^T V^T M V y}. \tag{6.4}$$

Making λ_R stationary relative to arbitrary variations in the m elements of y yields the reduced eigenproblem

$$V^T K V y = \lambda_R V^T M V y. \tag{6.5}$$

We can view this reduction as imposing $n-m$ constraints on the original system, thus giving the following result using the Cauchy Interlace Theorem [30].

$$\lambda^{(i)} \leq \lambda_R^{(i)} \leq \lambda^{(i+n-m)} \qquad i \leq m. \tag{6.6}$$

Thus the ith approximation $\lambda_R^{(i)}$ is contained between $\lambda^{(i)}$ and $\lambda^{(n)}$ and the approximations become exact for $m = n$.

The essence of the reduction scheme lies in the definition of the transformation matrix V. Some researchers have used matrices comprised from vectors that span a Krylov subspace [44], [64]; we choose to use a matrix that is made from exact modal vectors [66]. By using the exact modal vectors, we use trial functions that are similar to the actual eigenvectors under small perturbations.

6.2 Methods

In the Rayleigh-Ritz method, the shape of deformation of the continuous system, $v(x)$ is approximated using a trial family of admissible functions that satisfy some geometric boundary condition of the problem

$$v(x) = \sum_{i=1}^{n} c_i \phi_i(x) \tag{6.7}$$

where c_i are unknown constant coefficients and ϕ_i are the known (or selected) trial family of admissible functions.

The accuracy of the method depends on the value of n and the choice of trial functions $\phi_i(x)$ used in the approximation. By using more elements in the sum, n, the approximation can be made more accurate. Similarly, by using trial functions which are close to the true eigenfunctions, the approximation can be improved. Again, by using actual eigenvectors, we utilize the best guess to the current solution by using information from a previous solution.

If we use this approximation technique to guess the vectors forming the solution to the eigenvalue problem in Equation 6.5, we have

$$y = \sum_{\alpha=1}^{n} q_\alpha U_n \tag{6.8}$$

or $y = Uq$ where $U = [U_1 U_2, \ldots, U_n]$. Substituting into Equation 6.5 we have

$$U^T K U q = \lambda_r U^T M U q. \tag{6.9}$$

In this form, one can see why using a subspace formed of eigenvectors of similar systems will generate an accurate approximation for the solution to the original system. We use this form to approximate the solution as the geometry changes.

6.2.1 Approximations from a Subspace

Let s denote a geometric parameter. For a given finite element model, we have a generalized eigenvalue problem

$$(K(s) - \lambda(s)M(s))u(s) = 0, \tag{6.10}$$

where $K(s)$ is the stiffness matrix of the system and $M(s)$ is the mass matrix at the given state of the geometry, and $\lambda(s)$ and $u(s)$ are an eigenvalue and its corresponding eigenvector for the system.

If $w(s)$ is accurate to $O(h)$ as an estimate for $u(s)$, then

$$\mu(s) = (w(s)^* K(s) w(s))/(w(s)^* M(s) w(s)) \tag{6.11}$$

is accurate to $O(h^2)$ as an estimate for $\lambda(s)$. This is the accuracy boost we want to utilize.

Suppose that we have computed eigenpairs $(\lambda(s_0), u(s_0))$ and $(\lambda(s_1), u(s_1))$, and now want to compute the pair $(\lambda(s_2), u(s_2))$. Then we can use the initial approximation $\mu(s)$ drawn from a Rayleigh-Ritz approximation on the pencil

$$(U^* K(s_2) U, U^* M(s_2) U) \tag{6.12}$$

where $U = [u(s_0)\ \ u(s_1)]$ is a concatenation of previous eigenvectors (or if several of the lowest eigenvalues are desired, then simply replace $u(s_0)$ with $u_1(s_0), u_2(s_0)$, ... and $u(s_1)$ with $u_1(s_1), u_2(s_1)$, etc.). This concatenation simply populates the subspace with more eigenvectors from each previous shape.

For most systems, the first few natural frequencies and associated natural modes dominate the dynamic response, and the contribution of higher natural frequencies and the corresponding mode shapes is negligible.

If the variation to the parameter s is $O(h)$, then the accuracy in approximating eigenvector $u_i(s_2)$ by extrapolating through $u_i(s_0)$ and $u_i(s_1)$ should be $O(h^2)$. That is, the approximation is good through the linear term – and the accuracy of the eigenvalue approximation should be $O(h^4)$. More generally, if we use an invariant subspace computed at k points, we expect $O(h^k)$ accuracy in the eigenvector, and a corresponding $O(h^{2k})$ accuracy in the computed eigenvalue.

Therefore, by building a basis from n eigenvectors sampled at k locations in parameter space, we can predict the same n eigenvectors and the corresponding eigenvalues at nearby points. Using this method we can capture the behavior of the eigenvectors rotating as the geometry changes. Moreover by solving a smaller eigenproblem, we can reduce the time to compute the modal parameters of the new system in order to determine a subset of eigenvalues and eigenvectors.

In the next two sections we describe a means of transferring information from one geometry to another. In the first case, the finite element mesh remains constant so that the discretization over the domain is the same for each change in shape. In the second case, a new mesh is created so that the domain has two discretizations.

6.2.2 Variable Mapping

Given that the changes made to the geometry are parametric in nature, it is possible to map the geometries from step to step, and interpolate the fields of interest from the original to the current configuration.

Square Plate Example

In this example, we will use a triangular plate element to discretize the domain. For this element the degrees-of-freedom (DOFs) at each node are the out-of-plane displacement w, and the rotations about the in-plane principal axes θ_x and θ_y as defined in Kwon and Bang [49]. Figure 8.2 shows these DOFs.

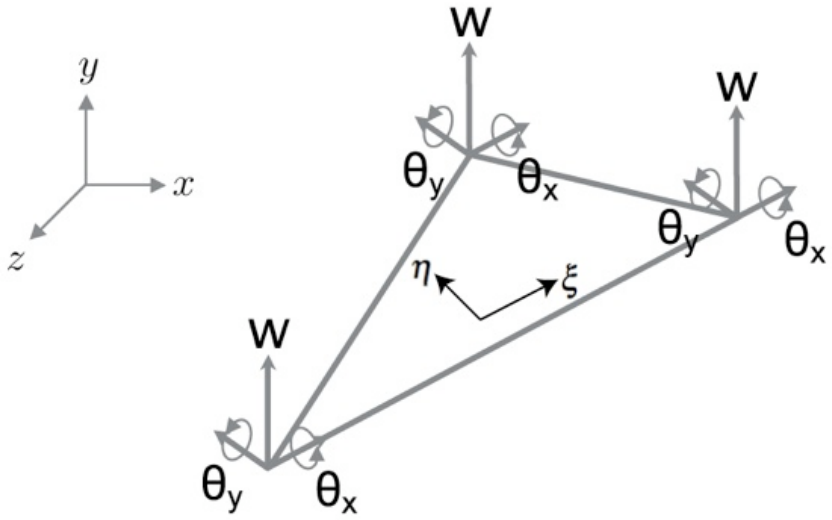

Figure 6.1. DOFs for a triangular plate element.

Let $\Omega_1, \Omega_2 \subset \mathbb{R}^2$ be two bodies (we think of them as the old and new configurations), and let $\phi : \Omega_1 \to \Omega_2$ be an invertible, differentiable map. Denote coordinates in Ω_1 with $\mathbf{X}$, and coordinates in Ω_2 with $\mathbf{x}$. Define also the deformation gradient $\mathbf{F} = \frac{\partial \phi}{\partial \mathbf{X}}$.

Suppose Ω_1 represents a plate, and $W(\mathbf{X})$ is the amount of vertical displacement of the plate. We assume there is no in-plane displacement. For three-node triangles, the nodal variables at each point would be

$$(W(\mathbf{X}), \theta_X(\mathbf{X}), \theta_y(\mathbf{X})) = \left(W, -\frac{\partial W}{\partial Y}, \frac{\partial W}{\partial X}\right)$$

We would like to move this information to Ω_2 using ϕ. That is, at a point $\mathbf{x} = \phi(\mathbf{X})$, we

want to know $w(\mathbf{x}) = w(\phi(\mathbf{X})) = W(\mathbf{X})$ and its derivatives:

$$(w(\mathbf{x}), \theta_x(\mathbf{x}), \theta_y(\mathbf{x})) = \left(w, -\frac{\partial w}{\partial y}, \frac{\partial w}{\partial x}\right).$$

The appropriate mapping in this case is $\nabla_{\mathbf{x}} = \mathbf{F}^{-T}\nabla_{\mathbf{X}}$. That is, the rotational degrees of freedom map according to

$$\begin{bmatrix} \theta_y \\ -\theta_x \end{bmatrix} = \mathbf{F}^{-T} \begin{bmatrix} \theta_Y \\ -\theta_X \end{bmatrix}.$$

If $J = \det(\mathbf{F})$, then we find

$$\begin{bmatrix} \theta_x \\ \theta_y \end{bmatrix} = J^{-1}\mathbf{F} \begin{bmatrix} \theta_X \\ \theta_Y \end{bmatrix}.$$

So in order to move the nodal variables (W, θ_X, θ_Y) at $\mathbf{X}_i$ to nodal variables (w, θ_x, θ_y) at a corresponding point $\mathbf{x}_i$, we need to multiply through by T_i, where

$$T_i := \begin{bmatrix} 1 & 0 \\ 0 & J^{-1}\mathbf{F} \end{bmatrix}.$$

Mapping Two Geometries

Figure 6.2 shows a planar geometry at two sample points S_1 and S_2, where x_i, y_i represents the coordinates of the geometry at a point.

The map between these geometries can be defined with the following parametric relation:

$$\phi(x,y) = \left(x, y\frac{s_2}{s_1}\right) \tag{6.13}$$

where s_1 and s_2 represent the height parameter values at the two sample points S_1 and S_2.

To map the undeformed geometry (Figure 6.3 (left)) to the deformed geometry (Figure 6.3 (right)), we apply the map:

$$\vec{x} \;=\; \phi\vec{X} \tag{6.14}$$

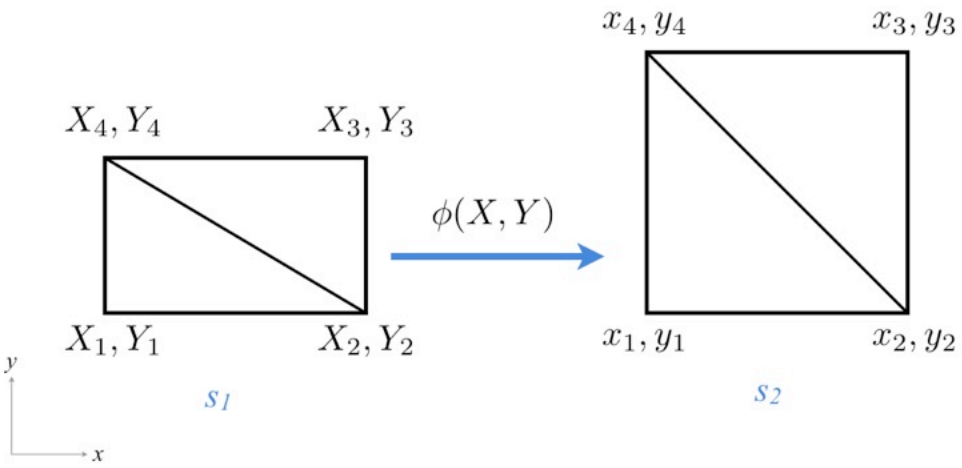

Figure 6.2. Left: Original geometry (S_1). Right: Deformed geometry (S_2).

where $\vec{x} = \{x\ y\}^T$ and $\vec{X} = \{X\ Y\}^T$ to each node in the undeformed geometry.

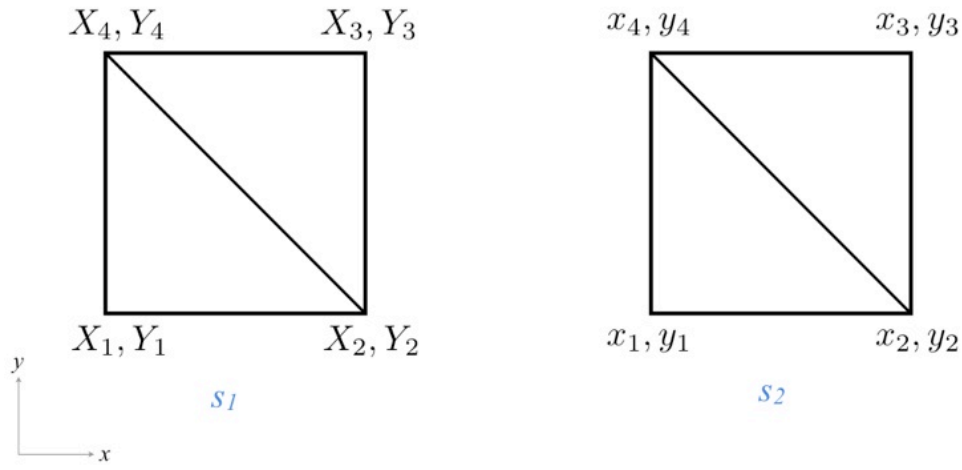

Figure 6.3. Left: Original geometry after map to cover the same domain as the deformed geometry(S_1). Right: Deformed geometry (S_2).

This mapping brings the domain covered by the first shape to be the same as the domain covered by the second shape.

Mapping Eigenvectors

Let the matrix $T = \mathbf{diag}(T_1, \ldots, T_n)$, where n is the number of nodes in the mesh. If the mesh we are using on Ω_2 is just a mapping of the mesh on Ω_1 under ϕ, then T is the map we should use to map displacement information (e.g. eigenvectors).

$$\vec{v}(x) = T\vec{V}(X) \tag{6.15}$$

If the meshes on the two domains are different, then there is an additional operator H that accounts for the interpolation, and the operator that we would ultimately use to move eigenvectors from one mesh to the other is the composition HT.

6.2.3 Remeshing

For large changes, the parametric deformation applied to mesh might distort the finite elements, in these cases, one would need to re-mesh the domain. Therefore an additional map must be applied to map the variables in the old mesh, to variables in the new mesh, essentially transferring variables from one mesh to another. This operation is similar to the geometric mapping, with the difference here being that the number of nodes in the new mesh need not be the same as the number of nodes in the old mesh (Figure 6.4).

To transfer variables in the old mesh to the variables in the new mesh, we interpolate the value using the finite element interpolation functions [75]. For each node in the new geometry, we find its surrounding element from the old geometry. This can be performed using a simple search.[1] Then inside this element, we find the local coordinates ξ and η using the element shape functions.

[1]For nodes that land on an edge, we pick either element. For nodes that land on a vertex, we pick any adjacent element.

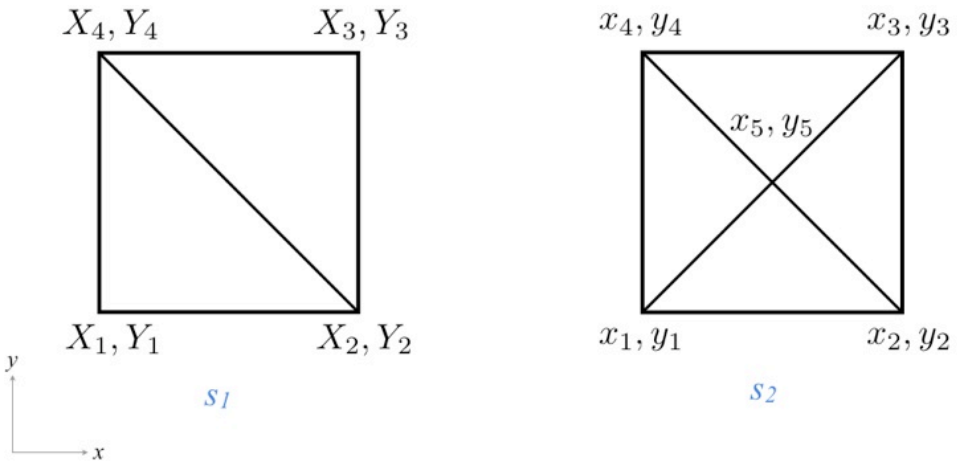

Figure 6.4. Left: Original geometry (S_1). Right: Deformed and re-meshed geometry (S_2).

For example, for a triangular element, we know that for any point

$$x(\xi,\eta) = \sum_{i=1}^{3} N_i(\xi,\eta)x_i \tag{6.16}$$

$$y(\xi,\eta) = \sum_{i=1}^{3} N_i(\xi,\eta)y_i \tag{6.17}$$

where

$$N_1(\xi,\eta) = 1-\xi-\eta \tag{6.18}$$

$$N_2(\xi,\eta) = \xi \tag{6.19}$$

$$N_3(\xi,\eta) = \eta \tag{6.20}$$

We then solve the system

$$\xi(x_2-x_1)+\eta(x_3-x_1)+(x_1-x_i) = 0 \tag{6.21}$$

$$\xi(y_2-y_1)+\eta(y_3-y_1)+(y_1-y_i) = 0 \tag{6.22}$$

$$\tag{6.23}$$

for ξ and η.

For the mapping from old mesh to new mesh, we then have

$$\begin{Bmatrix} \vec{x_1} \\ \vec{x_2} \\ \vec{x_3} \\ \vec{x_4} \\ \vec{x_5} \end{Bmatrix} = \begin{bmatrix} M(x_1) & M(x_1) & M(x_1) & M(x_1) \\ M(x_2) & M(x_2) & M(x_2) & M(x_2) \\ M(x_3) & M(x_3) & M(x_3) & M(x_3) \\ M(x_4) & M(x_4) & M(x_4) & M(x_4) \\ M(x_5) & M(x_5) & M(x_5) & M(x_5) \end{bmatrix} \begin{Bmatrix} \vec{X_1} \\ \vec{X_2} \\ \vec{X_3} \\ \vec{X_4} \end{Bmatrix} \tag{6.24}$$

where $M_j(x^i)$ is the shape function of the old element evaluated at the point in the new mesh.

We can rewrite Equation 6.24 as

$$\vec{x} = h\vec{X} \tag{6.25}$$

Recall that we are using this mapping to transfer the eigenvector information from one geometry to the next under deformation. Therefore, for each eigenvector of interest, we use the mapping matrix, H, to interpolate to the new system size.

$$\vec{v}(x) = HT\vec{V}(X) \tag{6.26}$$

where $H = h \otimes I$ expands the mapping to the number of degrees of freedom at each node.

We can apply this technique to the eigenvectors of the old system to transfer them to the new system. We then use the vectors as before, concatenating them to form a Ritz basis. In the next section we examine the accuracy of the technique on several examples to see how well we can approximate the resonant frequencies after modifications to the geometry.

6.3 Results

We tested the usefulness of this approach on a variety of experiments. The geometries tested do not represent full instruments per-se, but instead are arbitrary shapes that can be formed using a parametric method. We use these parametric shapes for examination of the method.

First we examine using the eigenvectors from the previous iterations directly, i.e. no geometric remapping. We then examine using geometric remapping to "warp" the eigenvectors from the old geometry to the new geometry. Finally, we examine using geometric re-meshing to transfer eigen-information from one mesh to another.

6.3.1 Square

We examined changing the height of a 1 meter tall $\times$ 1 meter wide plate [2] by 10 centimeters, 1 centimeter, and 1 millimeter and examined the error in the prediction of the new resonant frequencies of the system. The geometry shown in Figure 6.5 uses 44 triangular plate elements and Figure 6.6 shows the deformation for the three step sizes.

No Geometric Mapping

When simply concatenating the previous eigenvectors, the approximation to the actual eigenvalues are shown in Figure 6.7 for various changes in square height. We considered using only one sample point in parameter space (i.e. one previous shape), s, and examined the accuracy in prediction of the eigenvalues compared to those computed by direct eigen-decomposition. The results show prediction errors of $4.77 \times 10^{-5}\%$ for the smallest step size

[2] The actual dimensions of the object do not affect the relative errors reported. The size of the object only affects the absolute error.

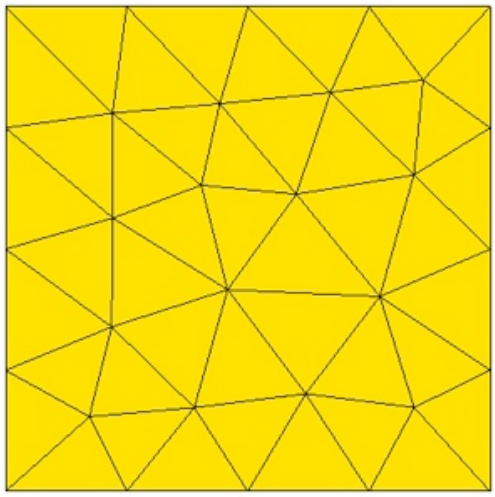

Figure 6.5. Simple plate model.

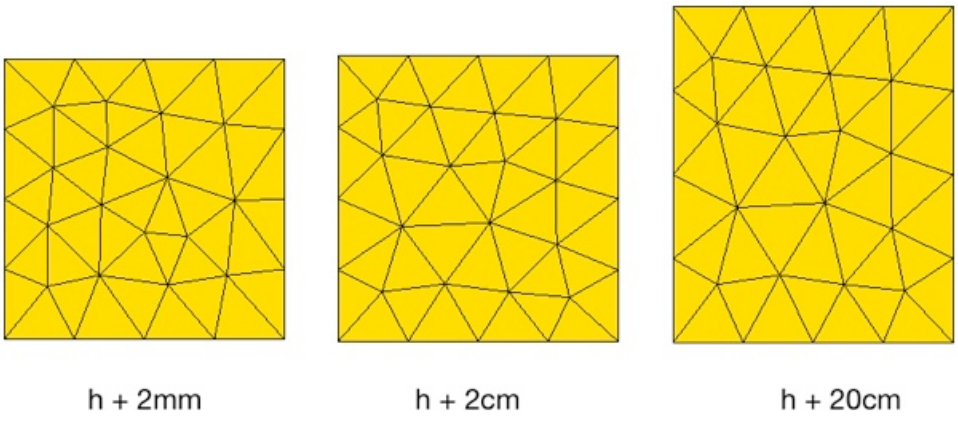

Figure 6.6. Simple plate model deformed.

and 7.75% for the largest step size. As expected, smaller changes in geometry allowed for

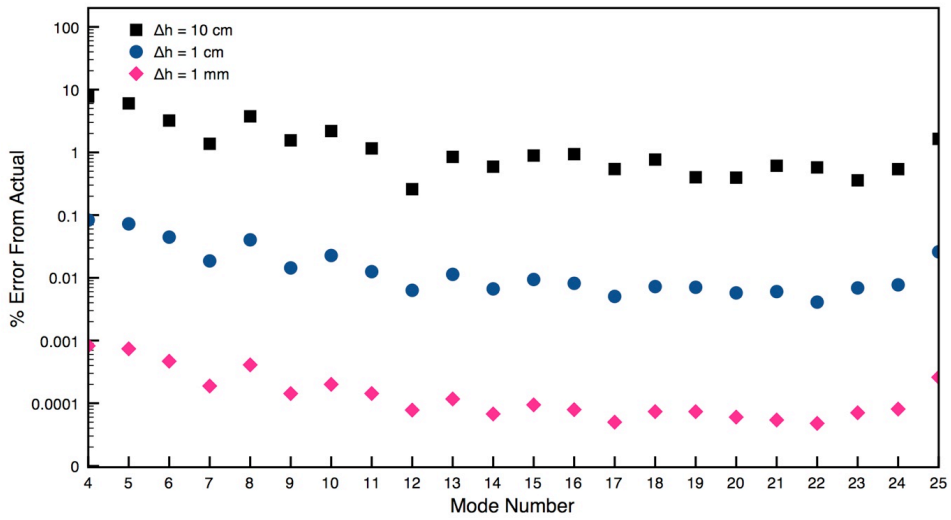

Figure 6.7. Error in prediction of resonant frequency from actual for different size Δh. One sample point.

better prediction of the new eigensolution. In each of the plots, we consider the first 25 non-zero eigenvalues.

Next, we examined using two sample points (i.e. two previous shapes). Figure 6.8 shows the results for the different step sizes. The results show prediction errors of 5.66×10^{-8}% for the smallest step size and 0.24% for the largest step size compared to resonant frequencies computed by direct eigendecomposition. As expected, using more points in parameter space increased the accuracy of the predictions. In fact, the accuracy of the two-subspace version is almost twice as many digits as the one-subspace version, which agrees with the theoretical bounds discussed previously.

Figure 6.9 shows a comparison between the predicted and the actual resonant frequencies for an overall 20 centimeter change (two sample points each making a 10 centimeters change) in height. These results show that the approximations are so close that the two values overlap.

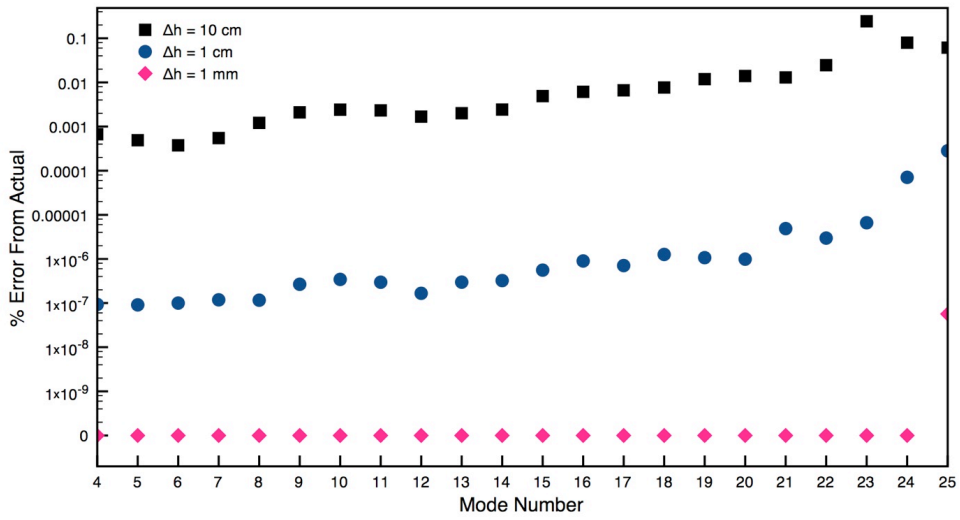

Figure 6.8. Error in prediction of resonant frequency from actual for different size Δh. Two sample points.

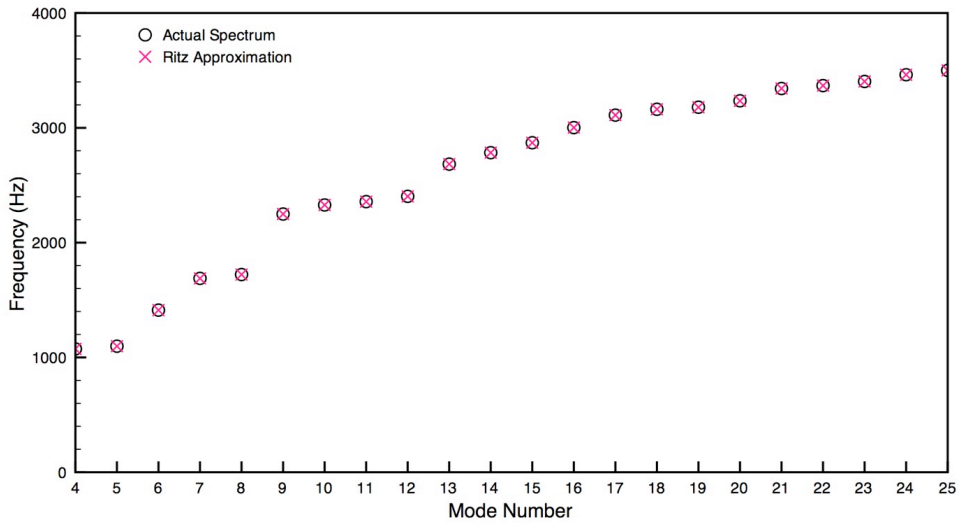

Figure 6.9. Comparison of predicted and actual resonant frequencies for square geometry, Δh = 10 cm.

Figure 6.10 shows that using two sample points versus one also gives much faster convergence. Notice how the two-point version has a steeper slope than the one-point version, following the expected $O(h^{2k})$ convergence as explained in Section 6.2.1, (where k is the number of points).

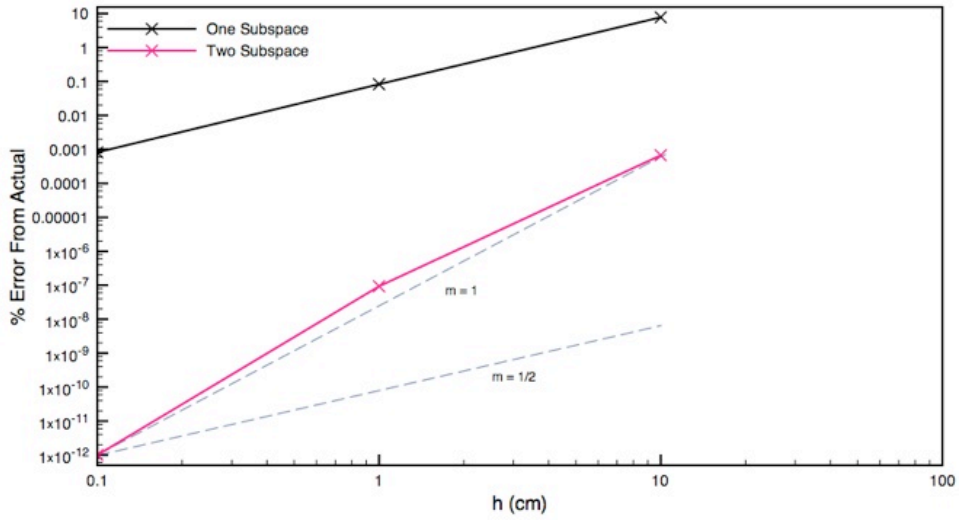

Figure 6.10. Convergence for different size Δh using one and two sample points.

We also investigated using a larger subspace. So instead of using the first 25 eigenvectors, we used the first 50. Figure 6.11 shows how using more eigenvectors from each of the two sample points improves the estimate of the eigenvalues. Error for the largest step size decreased to $3.95 \times 10^{-3}\%$ for the first 25, and $2.65 \times 10^{-2}\%$ for all 100 eigenvalues.

Geometric Mapping

If we remap the eigenvectors from the old geometry to the new using Equation 6.15, we see the following approximations (Figure 6.12) when using two sample points. In this example, the number of nodes in the mesh remained constant as no re-meshing was performed. These results show that using geometric remapping preserves the minimum error

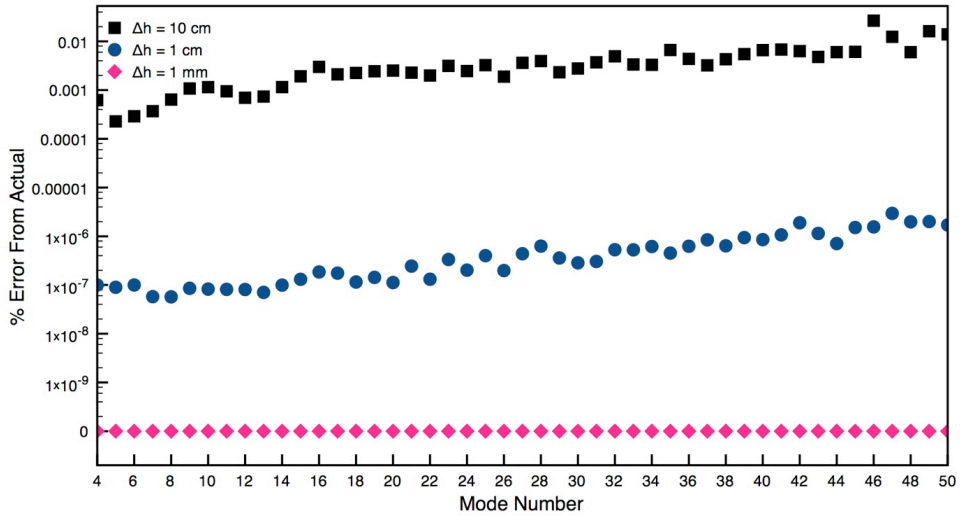

Figure 6.11. Error in prediction of resonant frequency from actual for Δh=10 cm. Larger subspaces.

and improves the maximum error to 0.15% (from 0.24%) for the largest step size. It should be noted that this example is the simplest test of this method. That is, since an eigenvector times a scale factor is also an eigenvector, we would expect a good mapping using this simple geometric deformation. In the remeshing and more complex geometries we see that the method works for non-trivial mappings.

Remeshing

Figure 6.13 shows the square geometry after deformation using re-meshing. Notice that for each of these shapes, the number of elements in the mesh is not constant.

For the cases where the geometry needs re-meshing, we can map the eigenvectors from the old geometry to the new using Equation 6.26. Figure 6.14 shows the error in approximations using two sample points. These results show that even for changes of up to 20% of the original object size, it is possible to predict the resulting frequency spectrum to within 11% error. This means that instead of performing a time-consuming reanalysis, one can

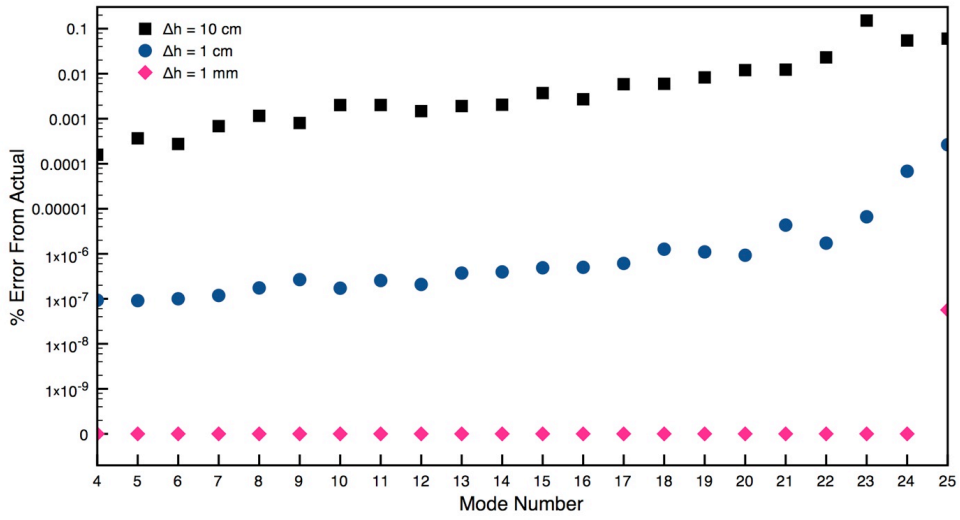

Figure 6.12. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric mapping.

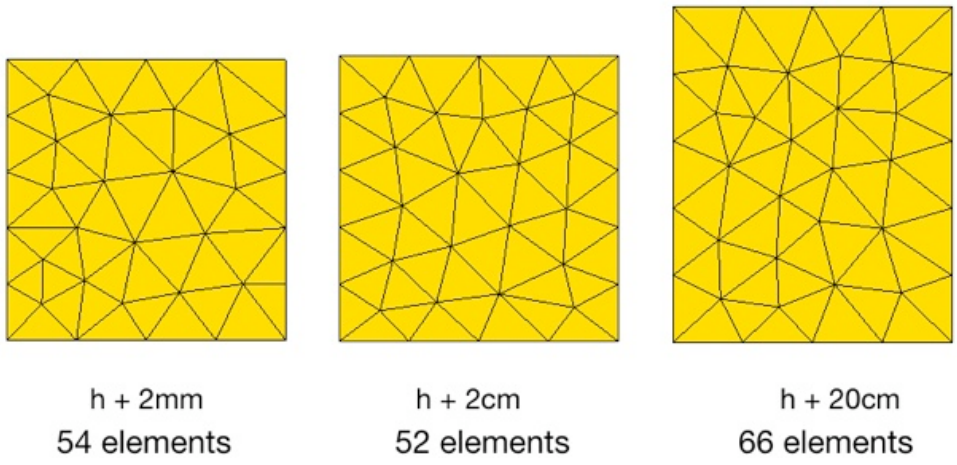

Figure 6.13. Simple plate model deformed and re-meshed.

make a reasonable estimate to the new spectrum even for large changes in geometry. Notice that re-meshing breaks the previous convergence relationships and that the plots do not strictly follow the $O(h^2k)$ relation, hence the closer spacing between the approximations. In

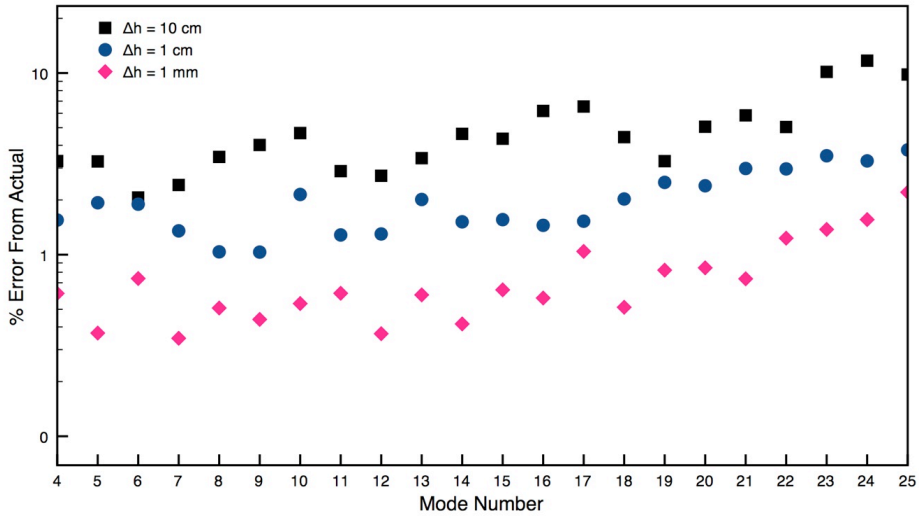

Figure 6.14. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric re-meshing.

this example we see that in the cases where the entire mesh changes, the interpolation is not as accurate as when the mesh is left intact and the geometry is remapped.

6.3.2 Shaped Plate

Next we tested this method on a more complicated plate shown in Figure 6.15. This plate is 2 meter × 0.5 meter with a elliptical cut on one side and is made up of 106 plate elements.

As the height of the plate changes, the profile of the elliptical cut changes as seen in Figure 6.16.

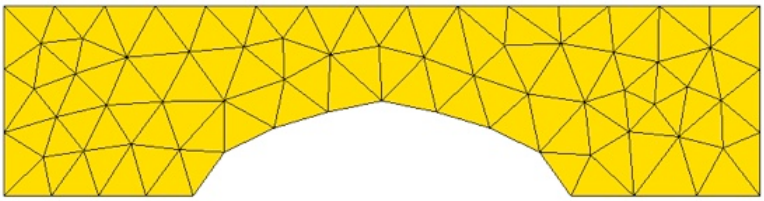

Figure 6.15. Curved plate model.

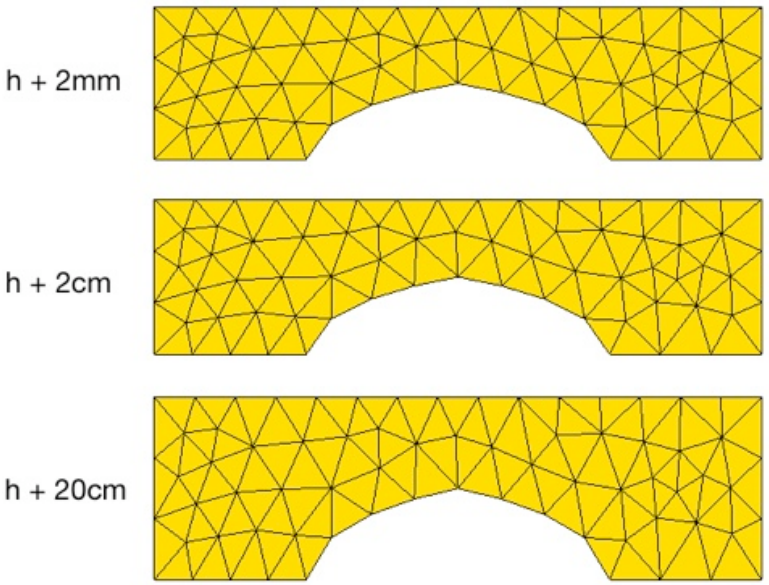

Figure 6.16. Curved plate model deformed.

No Geometric Mapping

When simply concatenating the previous eigenvectors, the approximation to the actual eigenvalues are shown in Figure 6.17 for various changes in square height using one sample point. These results show errors of 0.02% for the smallest and 134.10% for the largest step sizes. Using two sample points improved the predictive capabilities to 1.81×10^{-7}% for the smallest and 1.56% for the largest step size as seen in Figure 6.18. Again we see that using two sample points greatly improves the resonant frequency estimates. This behavior follows the convergence seen in Figure 6.10.

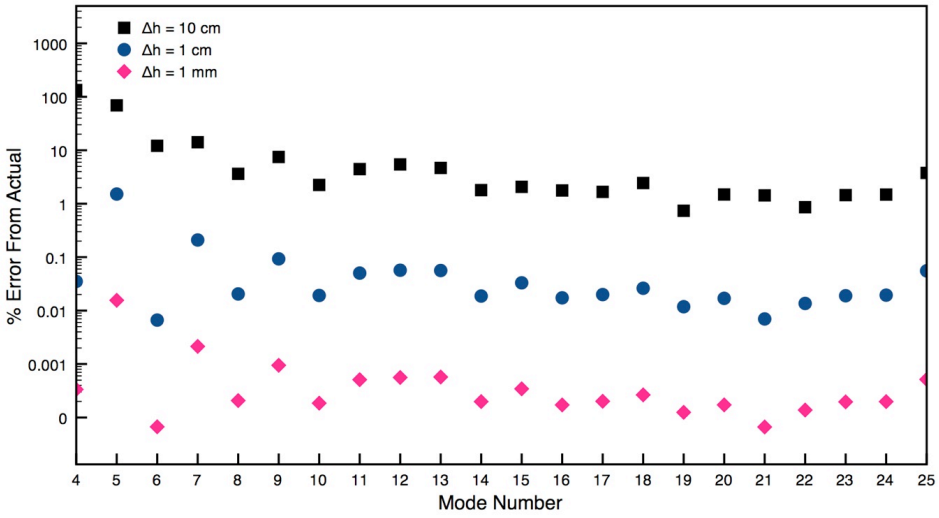

Figure 6.17. Error in prediction of resonant frequency from actual for different size Δh. One sample point.

Figure 6.19 shows a comparison of the actual and the predicted spectrum. This figure shows that using two sample points allows for an accurate prediction even with a 20% change in the object's height.

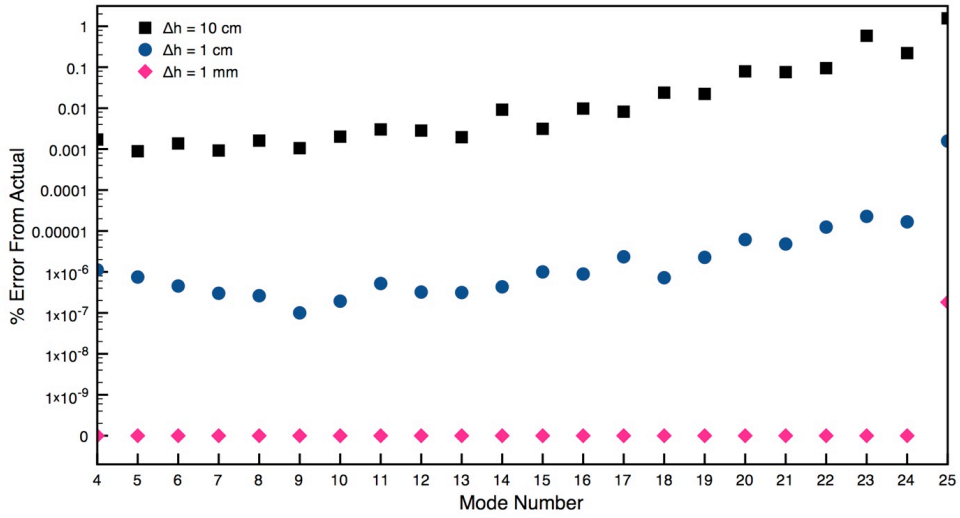

Figure 6.18. Error in prediction of resonant frequency from actual for different size Δh. Two sample points.

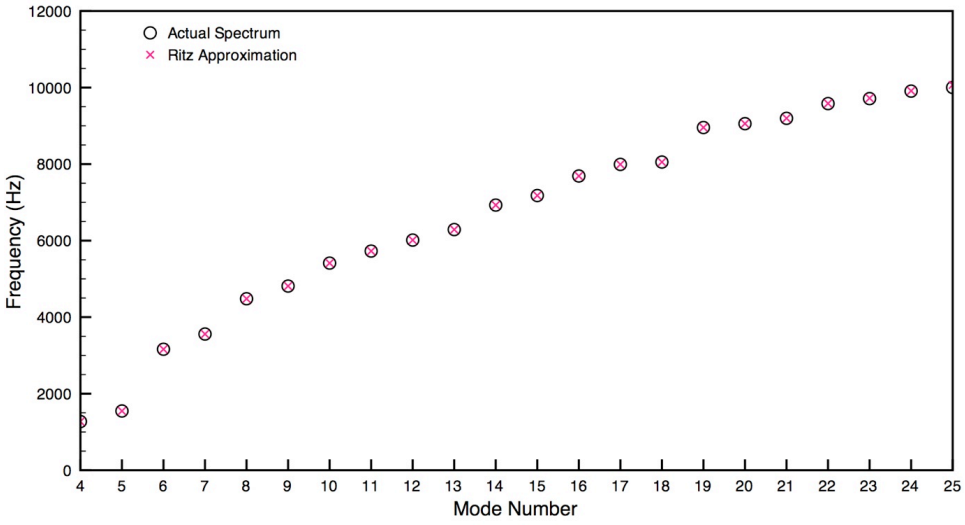

Figure 6.19. Comparison of predicted and actual resonant frequencies for the shaped plate, Δh = 10 cm.

Geometric Mapping

If we remap the eigenvectors from the old geometry to the new using Equation 6.15, the approximation improves the largest step size error to 0.88% from the previous 1.56% (Figure 6.20).

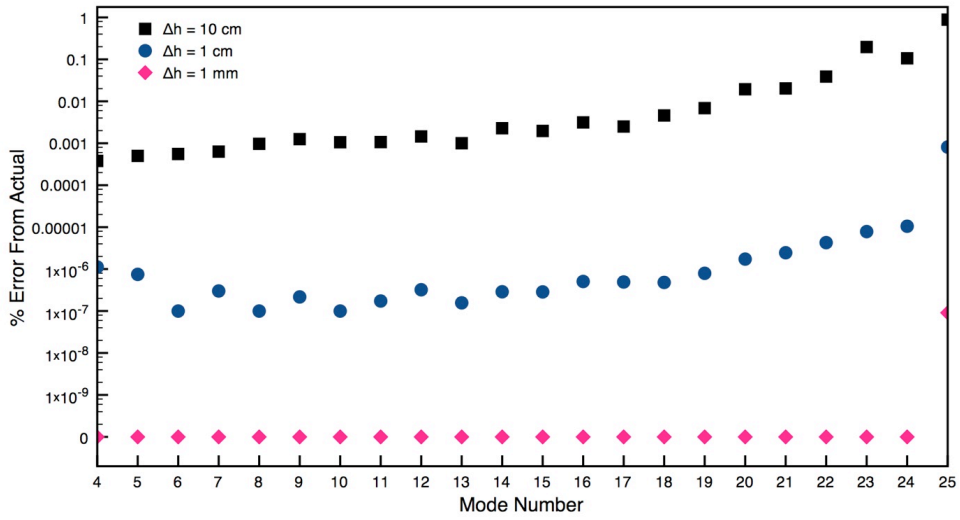

Figure 6.20. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric mapping.

Remeshing

Figure 6.21 shows the curved plate geometry after deformation using re-meshing. Notice that for each of these shapes, the number of elements in the mesh is not constant.

When re-meshing the geometry at each sample point, the prediction capabilities follow the curves shown in Figure 6.22. These results show a maximum error of 8.47% using geometric re-meshing. This means that even for a 20% change in the height of the object, we can still accurately predict the new spectrum.

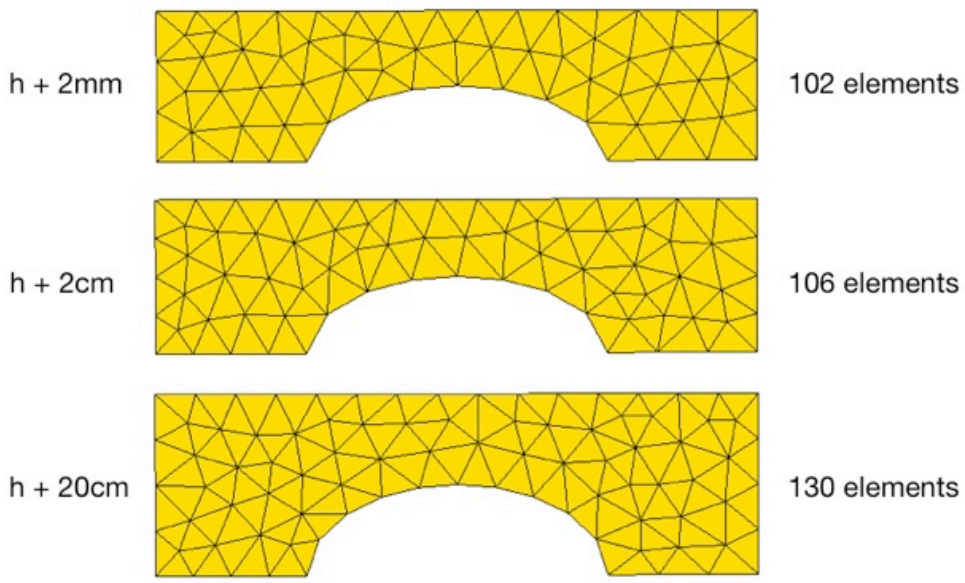

Figure 6.21. Curved plate model deformed and re-meshed.

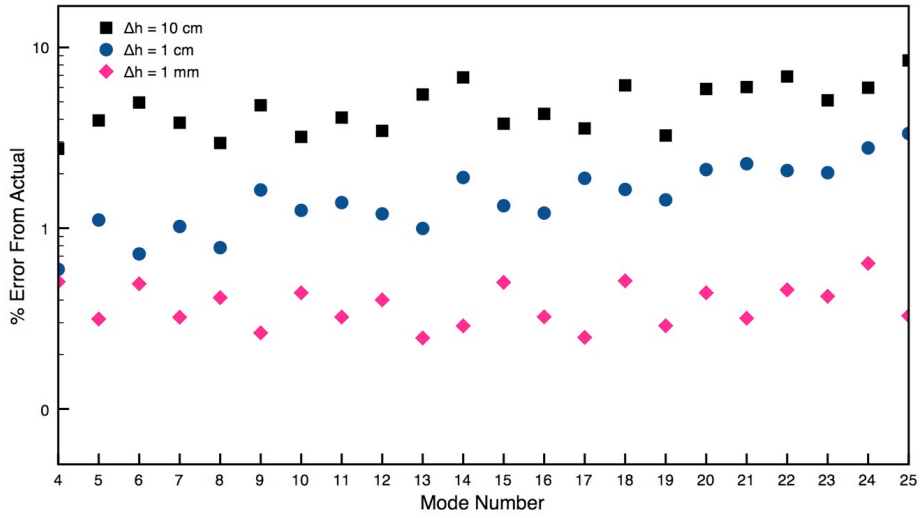

Figure 6.22. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric re-meshing.

6.3.3 Marimba Bar

To test the approximation capability for 3D objects, we extruded the previous plate to form a solid marimba bar (Figure 6.23). This object is 2 meter × 1 meter × 1 meter and is made of 619 tetrahedral elements as described in Bhatti [17].

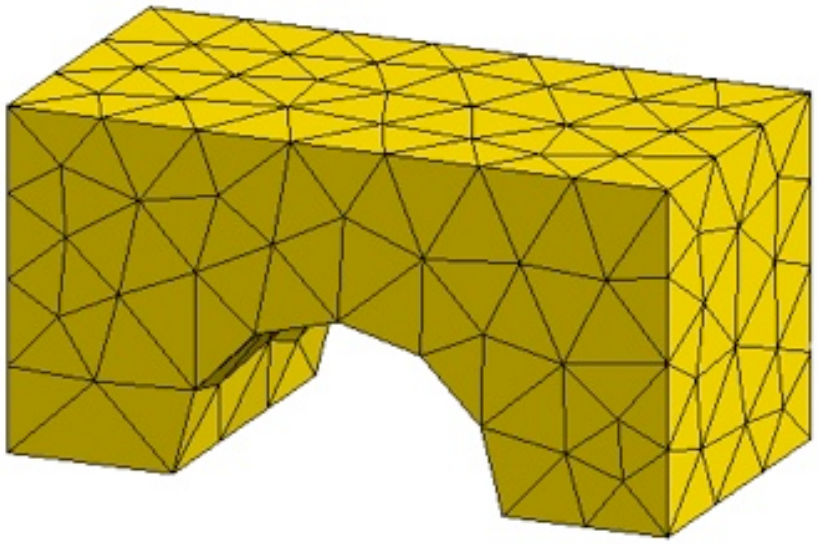

Figure 6.23. Tetrahedral solid model.

As the height of the block changes, the profile of the elliptical cut changes as seen in Figure 6.24.

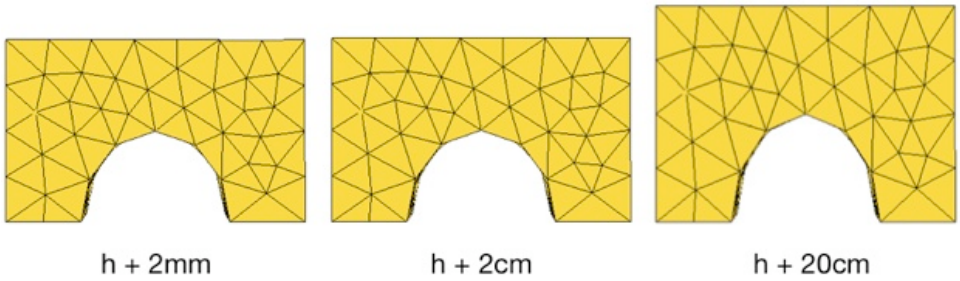

Figure 6.24. Curved block model deformed.

No Geometric Mapping

We examine the error when changing the height of the object by 10 centimeters, 1 centimeter, and 1 millimeter using two sample points. Figure 6.25 shows the maximum error is 0.40% and for a 10 centimeters change in geometry. Figure 6.26 again shows good prediction of the frequency spectrum.

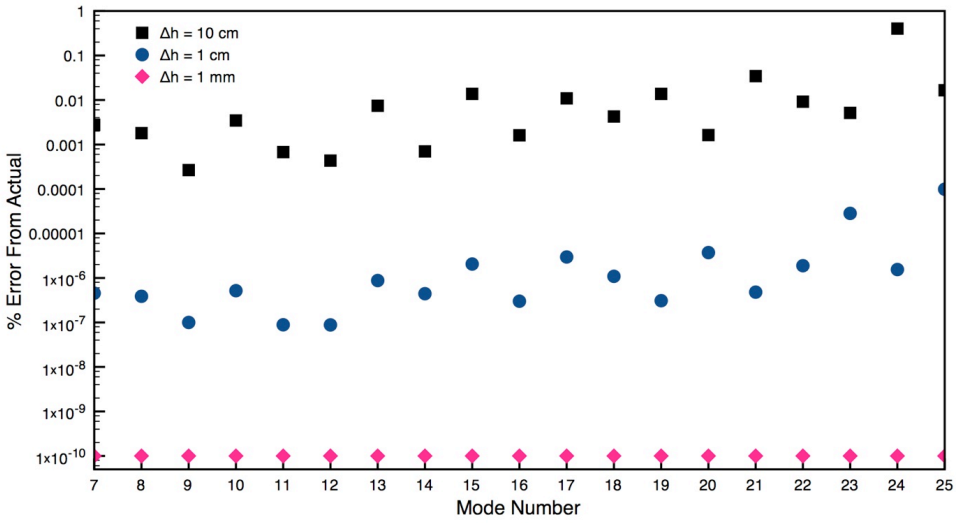

Figure 6.25. Error in prediction of resonant frequency from actual for different size Δh. Two sample points.

Geometric Mapping

Using geometric mapping and two sample points, the maximum error decreases to 0.38% (Figure 6.27). It is interesting to notice that for both the single and two point approximation the prediction follows a pattern of oscillating accuracy.

This oscillating behavior means that some eigenvectors are mapping to the new domain more accurately than others. For example, for the tenth eigenvector (which has a higher error than say the ninth eigenvector) we see the following mapping (Figure 6.29). What

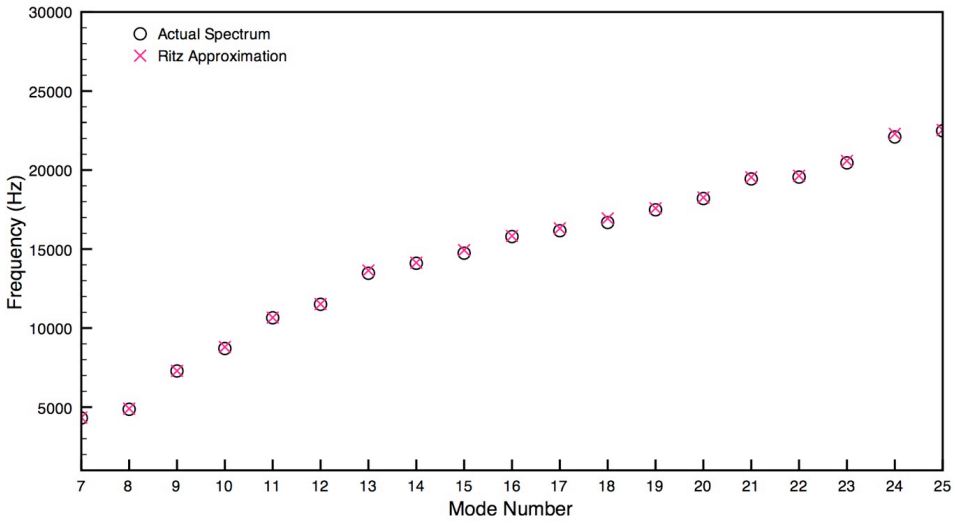

Figure 6.26. Comparison of predicted and actual resonant frequencies for solid model, Δh = 10 cm.

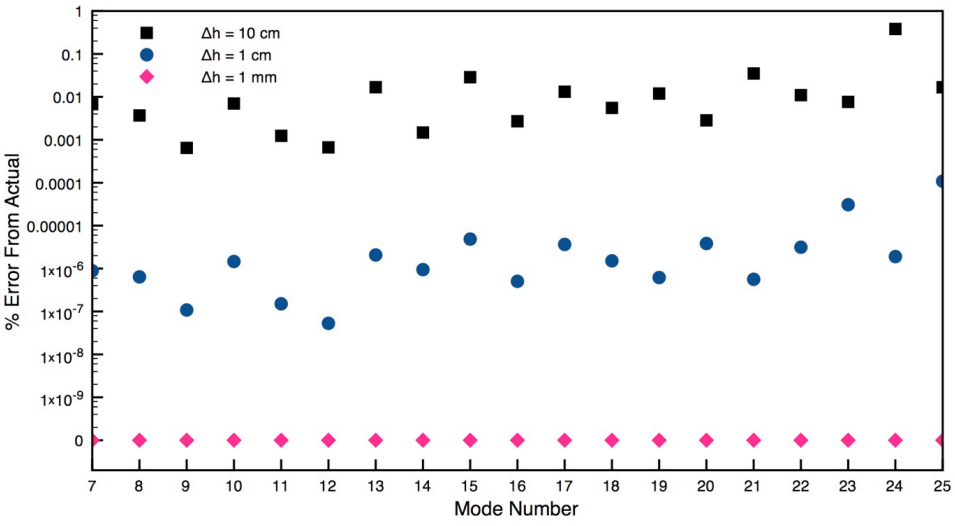

Figure 6.27. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric mapping.

this figure shows is that after remapping, the original geometry maps the deformed domain exactly (Figure 6.28), but the eigenvectors only approximate the deformed eigenvectors, i.e. it is not an exact map. For the ninth eigenvector however, the mapping is much closer (Figure 6.30).

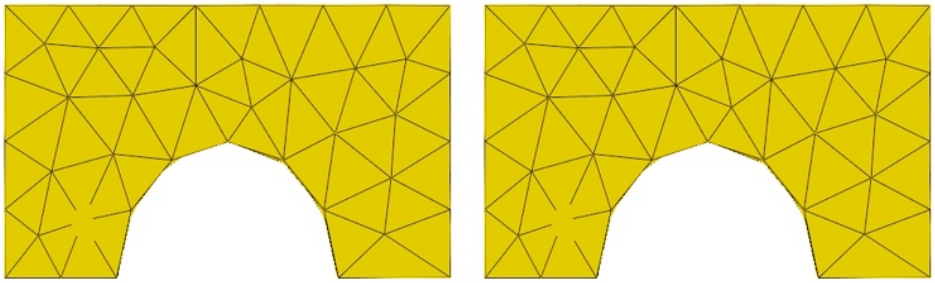

Figure 6.28. Left: The deformed geometry. Right: After remapping the initial geometry.

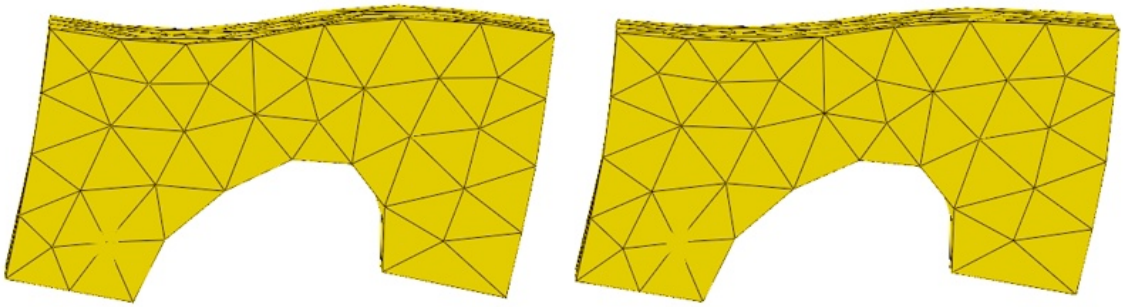

Figure 6.29. Left: The tenth eigenvector in the deformed geometry. Right: The tenth eigenvector after remapping the original geometry.

Figure 6.31 shows the visual overlay of the original geometry mapped to the deformed geometry for the tenth mode. Notice how this mapping only approximates the final domain and thus does not cover the new domain.

Figure 6.32 shows the visual overlay of the original geometry mapped to the deformed geometry for the ninth mode. Notice how this mapping allows the original geometry to completely fill the new domain.

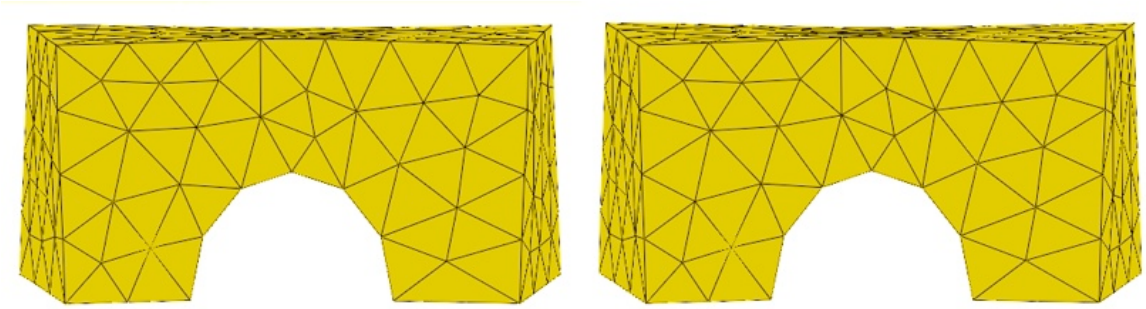

Figure 6.30. Left: The ninth eigenvector in the deformed geometry. Right: The ninth eigenvector after remapping the original geometry.

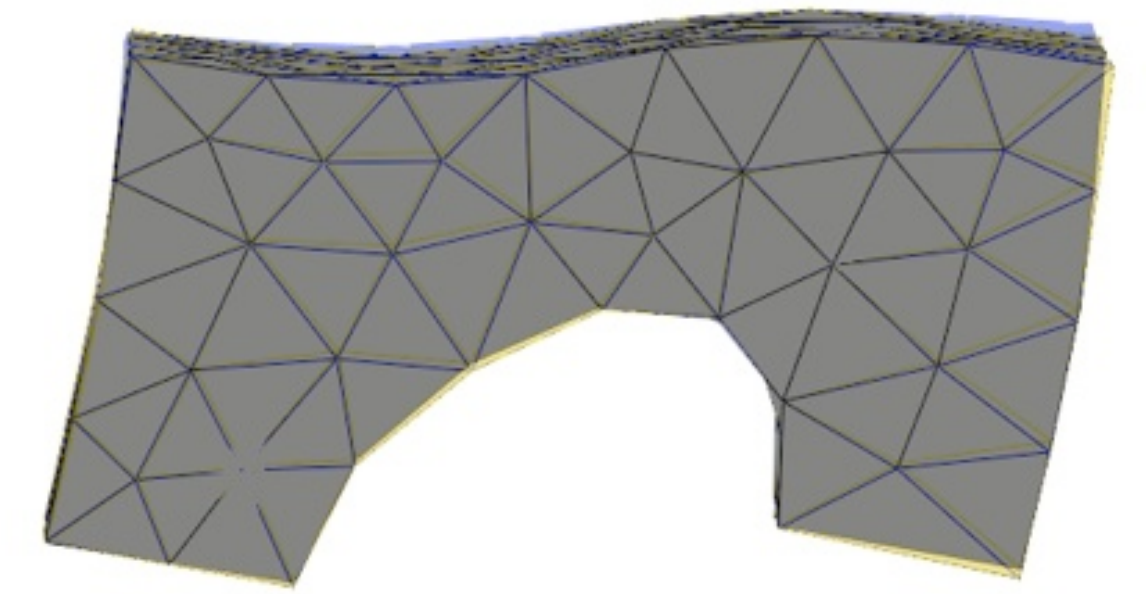

Figure 6.31. The tenth eigenvector in the deformed geometry mapped to the new geometry. The mapping is only approximate.

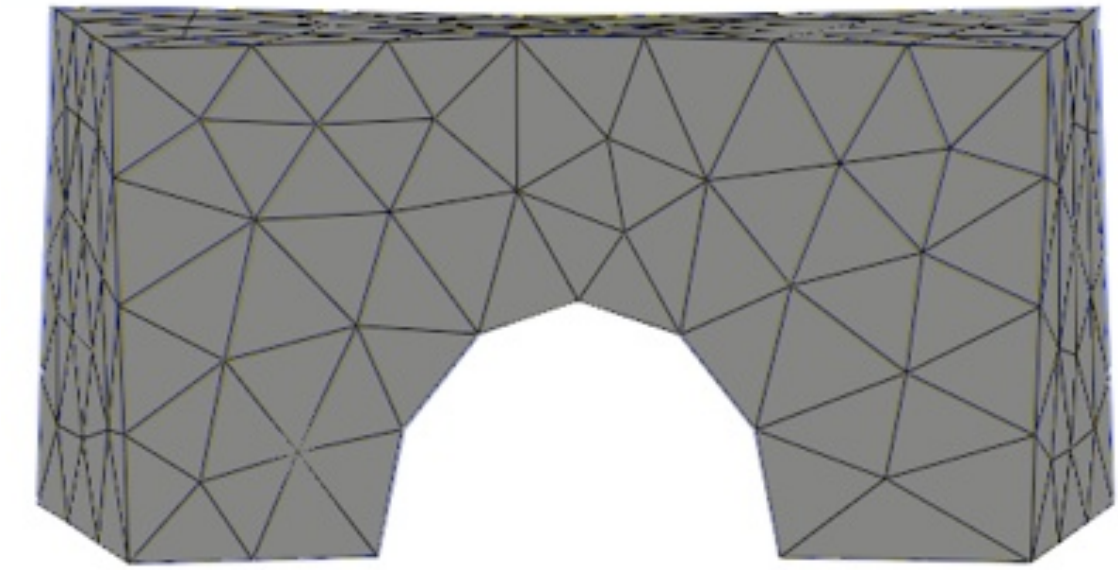

Figure 6.32. The ninth eigenvector in the deformed geometry mapped to the new geometry. The mapping is only approximate.

Remeshing

Figure 6.33 shows the curved block geometry after deformation using re-meshing. Notice that for each of these shapes, the number of elements in the mesh is not constant.

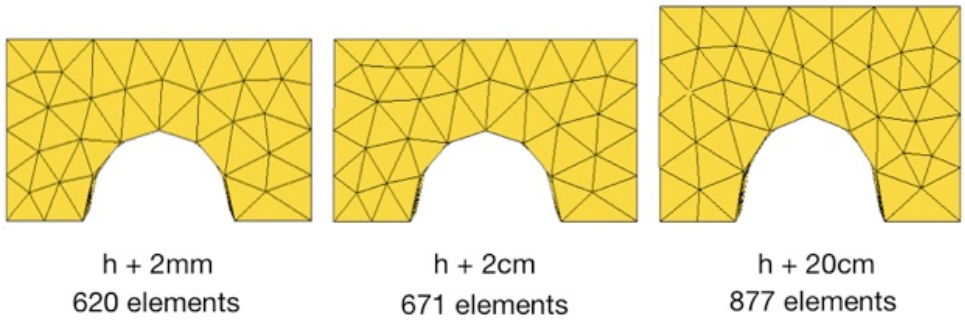

Figure 6.33. Curved solid model deformed and re-meshed.

When resmeshing the geometry, the maximum error in Figure 6.34 is 11.5%. Notice that for this example, the error is not necessarily only proportional to the step size. These

results suggest that another factor, such as mesh (and mode shape) similarity between steps can increase accuracy.

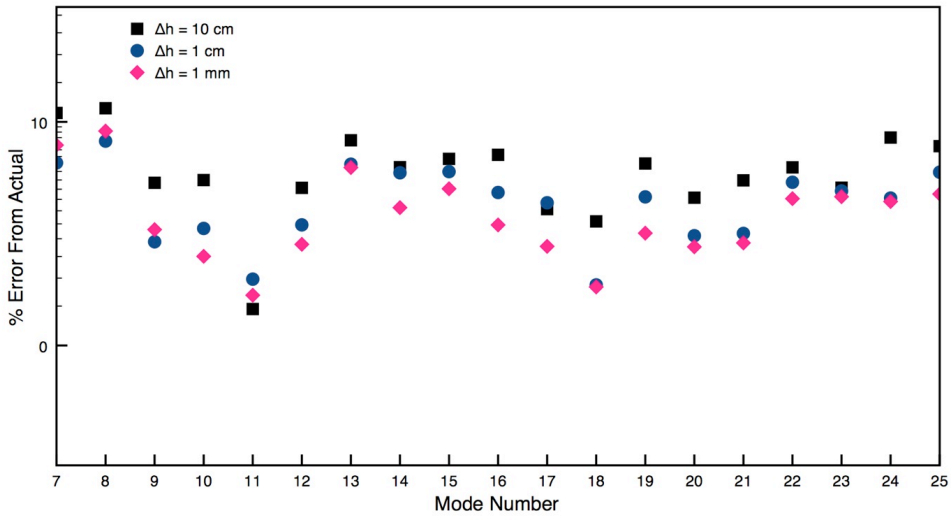

Figure 6.34. Error in prediction of resonant frequency from actual for different size Δh. Two sample points and geometric re-meshing.

6.3.4 Axisymmetric Geometry

Next we tested the parameterized geometry as described in Section 5.1.2. For this geometry, we used a linear shell finite element formulation as described in Kwon and Bang [49]. Each element consists of four nodes each with six degrees-of-freedom.

We examined a shell whose curvature is defined by four control points as shown in Figure 5.1. The crosses indicate the points modified directly. This curve is then divided by lateral points which are interpolated using uniform cubic B-spline interpolation.

The control points define a curve which is then revolved around the z-axis to form an axisymmetric geometry. By changing the location of these control points, we change the geometry parametrically. We examined changing a 1 meter tall, 1 meter radius object shown in Figure 6.35. We changed this object's outermost radius by 10 centimeters, 1 centimeter,

and 1 millimeter and examined the error in eigendecomposition. Figure 6.36 shows the

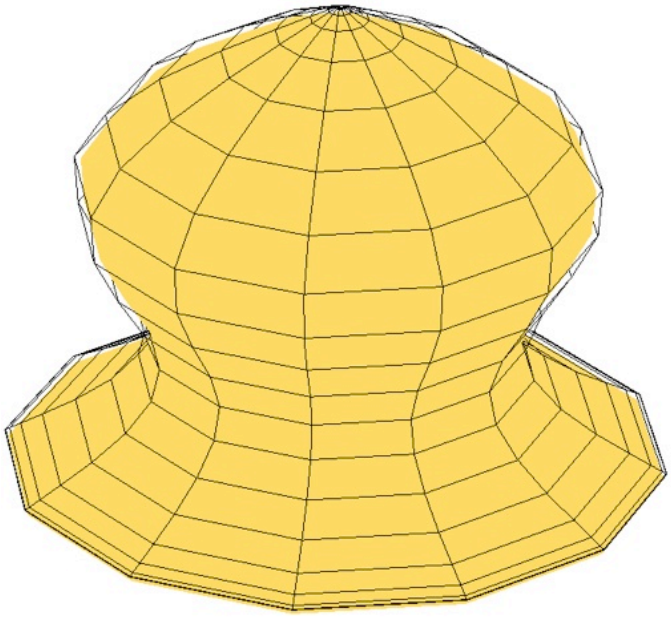

Figure 6.35. Simple shell model.

stages of deformation of the bell.

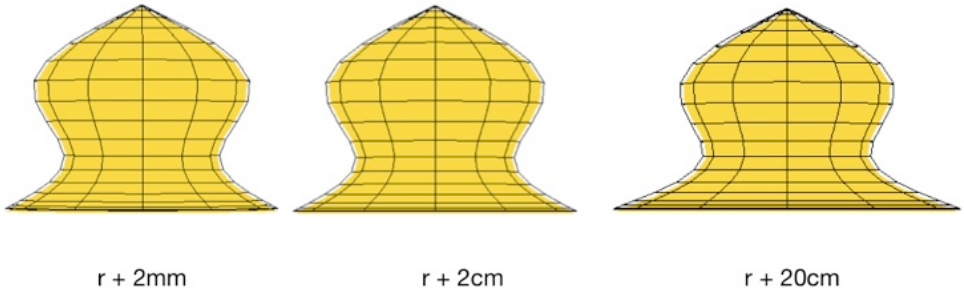

Figure 6.36. Bell model deformed.

No Geometric Mapping

We considered using only one sample point in parameter space, s, and examined the accuracy in prediction of the Ritz values. The results in Figure 6.37 show prediction errors of $4.20 \times 10^{-3}\%$ for the smallest step size and 30.2% for the largest step size compared to

those computed by direct eigendecomposition. Again, smaller changes in geometry allowed for better prediction of the new eigensolution.

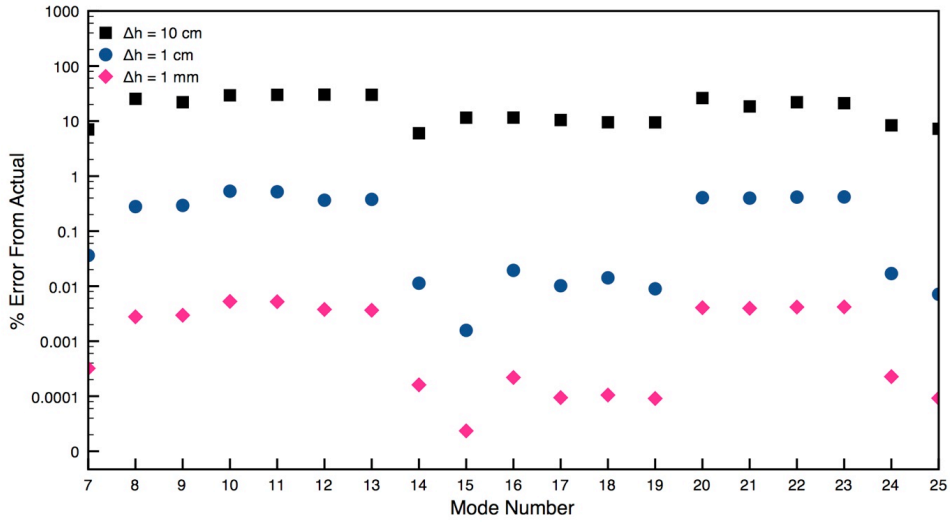

Figure 6.37. Error in prediction of resonant frequency from actual for different size Δh. One sample point.

Next, we examined using two sample points. Figure 6.38 shows the results for the different step sizes. Again using more points in parameter space increased the accuracy of the predictions by reducing the maximum error to 10.37%. Figure 6.39 shows again that using two sample points versus one also gives much faster convergence.

It is interesting to note that the axisymmetric bell has many more repeated eigenvalues than the previous models (Figure 6.40). We will discuss the possible ramifications of these repeated eigenvalues in the discussion section.

6.3.5 Performance

The speedup gained by using this method over traditional reanalysis is the difference between modest linear and super-linear computing time once the initial k locations in param-

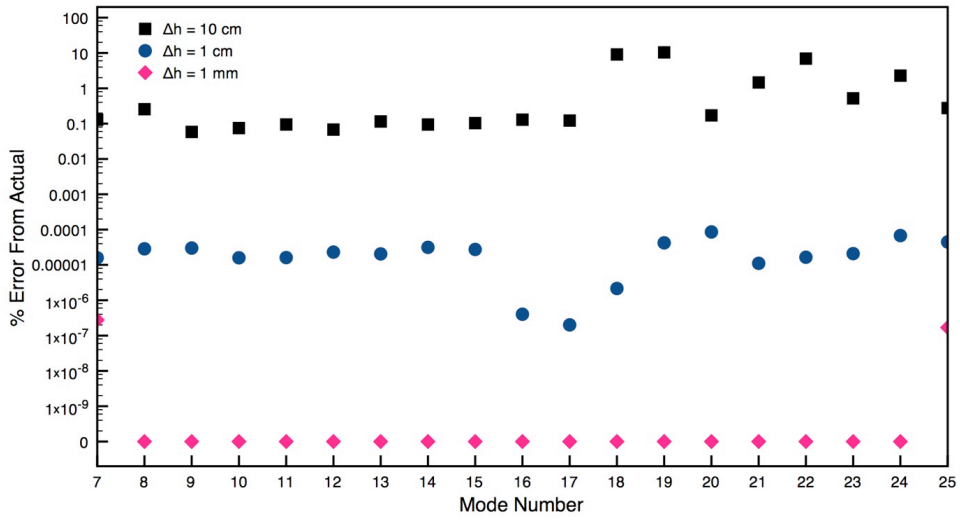

Figure 6.38. Error in prediction of resonant frequency from actual for different size Δh. Two sample points.

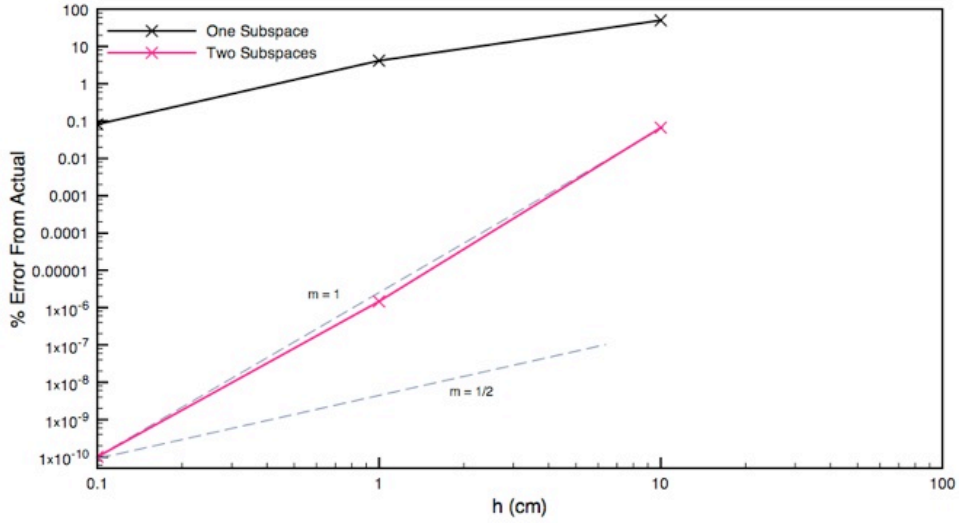

Figure 6.39. Convergence for different size Δh. Two sample points.

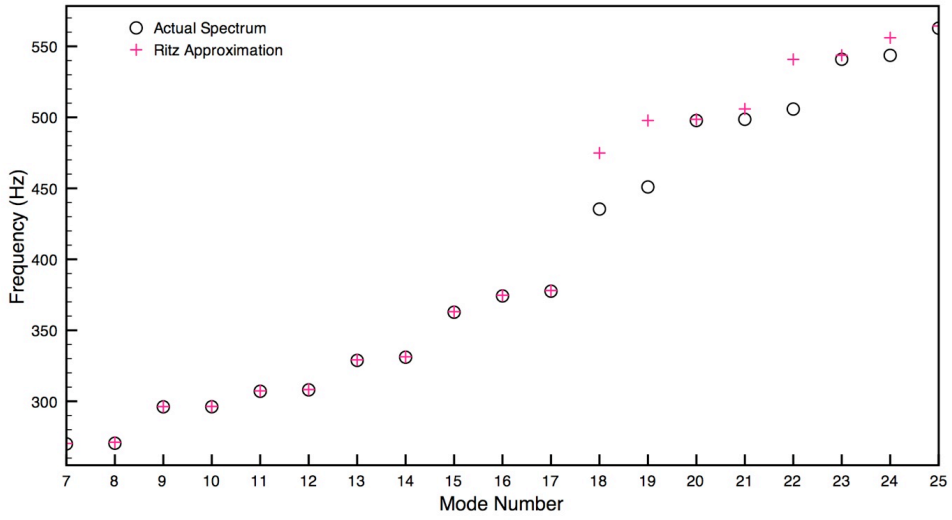

Figure 6.40. Comparison of predicted and actual resonant frequencies for the axisymmetric shell geometry, $\Delta h = 10$ cm.

eter space have been computed. Figure 6.41 shows the speedup using this method without re-meshing, over reanalysis for increasing resolution of the object shown in Figure 6.35.

6.3.6 Large Overall Shape Changes

By concatenating the approximations as small changes are made to the object, one can effectively track large changes. Figure 6.42 and 6.43 shows four steps in an interactive environment. As changes are made, the approximation (red dots) follows the true spectrum (black dots). These figures show even though the final shape is very different from the starting shape, the intermediate approximations aid in the approximations of the final shape.

6.4 Discussion

For a model with a high degree of rotational symmetry (such as in Figure 6.35), our test problem exhibits many repeated eigenvalues. Generally, for an eigenvalue with multiplicity

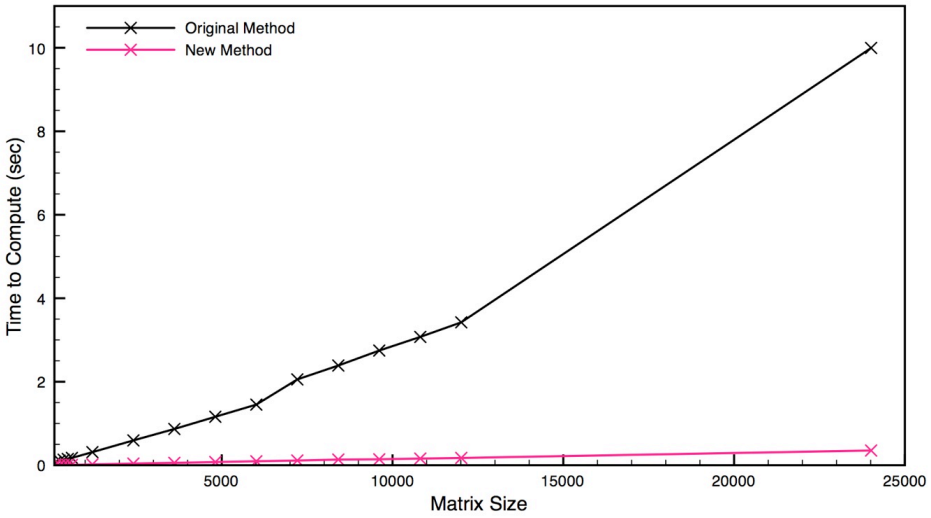

Figure 6.41. Time to compute new spectrum no re-meshing.

m, we cannot uniquely identify m mode shapes. Even for nearly-symmetric objects, the mode shapes associated with a cluster of eigenvalues can vary wildly under small perturbations. In these cases, typically, only the m-dimensional invariant subspace spanned by all the shapes for the eigenvalue cluster is uniquely defined.

The sensitivity of the mode vectors does not, on its own, imply that our method will behave poorly in the presence of repeated eigenvalues. If every vector in the invariant subspace for a cluster of m eigenvalues can be approximated well by some vector in the projection basis U, then we expect Rayleigh-Ritz approximation with U to produce a cluster of m eigenvalues near the original eigenvalues, and a corresponding subspace which is a good approximation to the true invariant subspace. However, the single-vector Lanczos iteration we use to find mode shapes sometimes fails to find a complete basis for each invariant subspace. When this occurs, we might overlook some of the eigenvalues and eigenvectors that we would like to capture in our projection space. The missing mode shapes would represent a significant failure in our method.

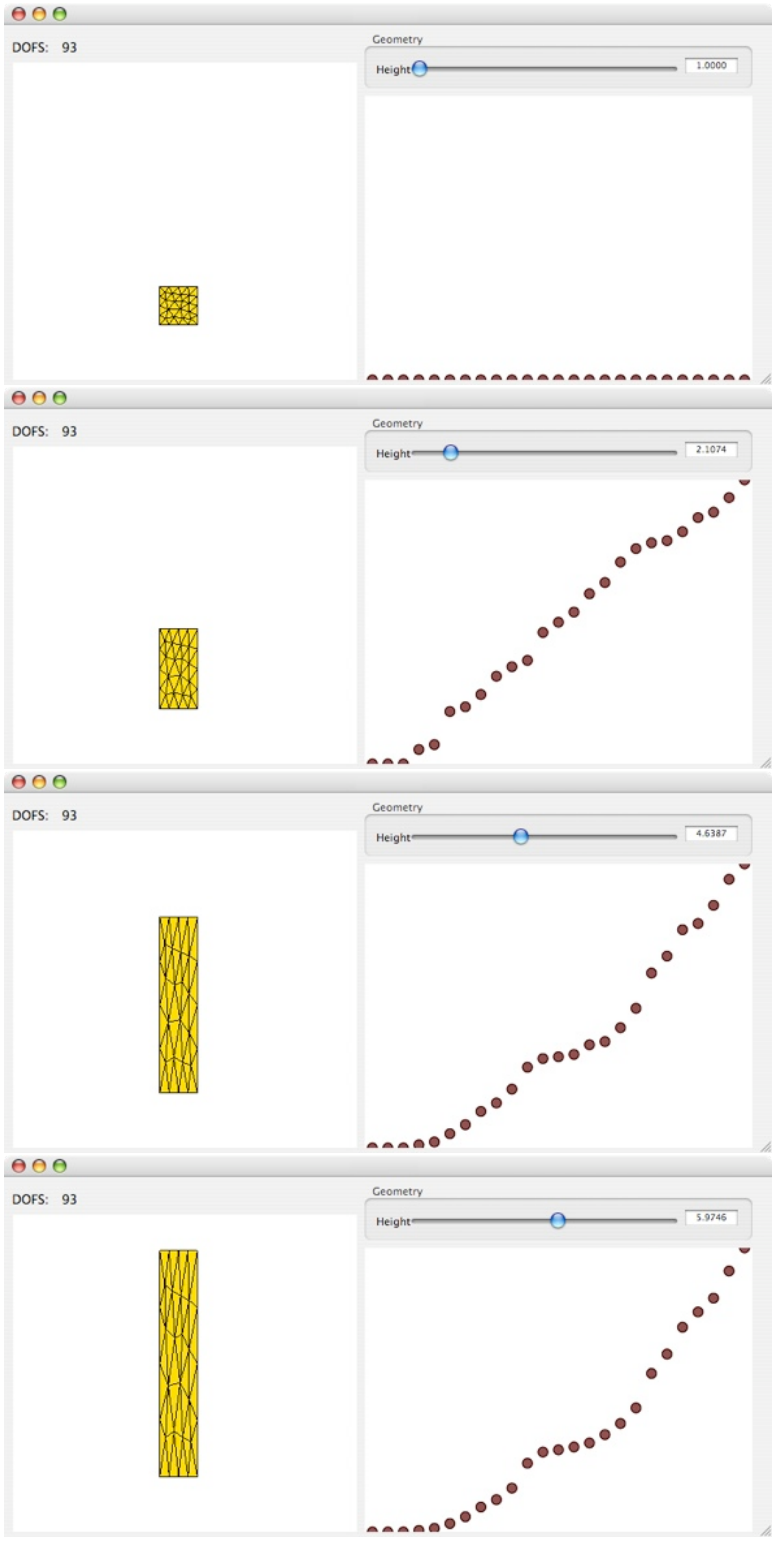

Figure 6.42. Four steps in an interactive shape-changing environment. Approximations to the spectrum (red dots) coincide exactly with the true spectrum (black dots).

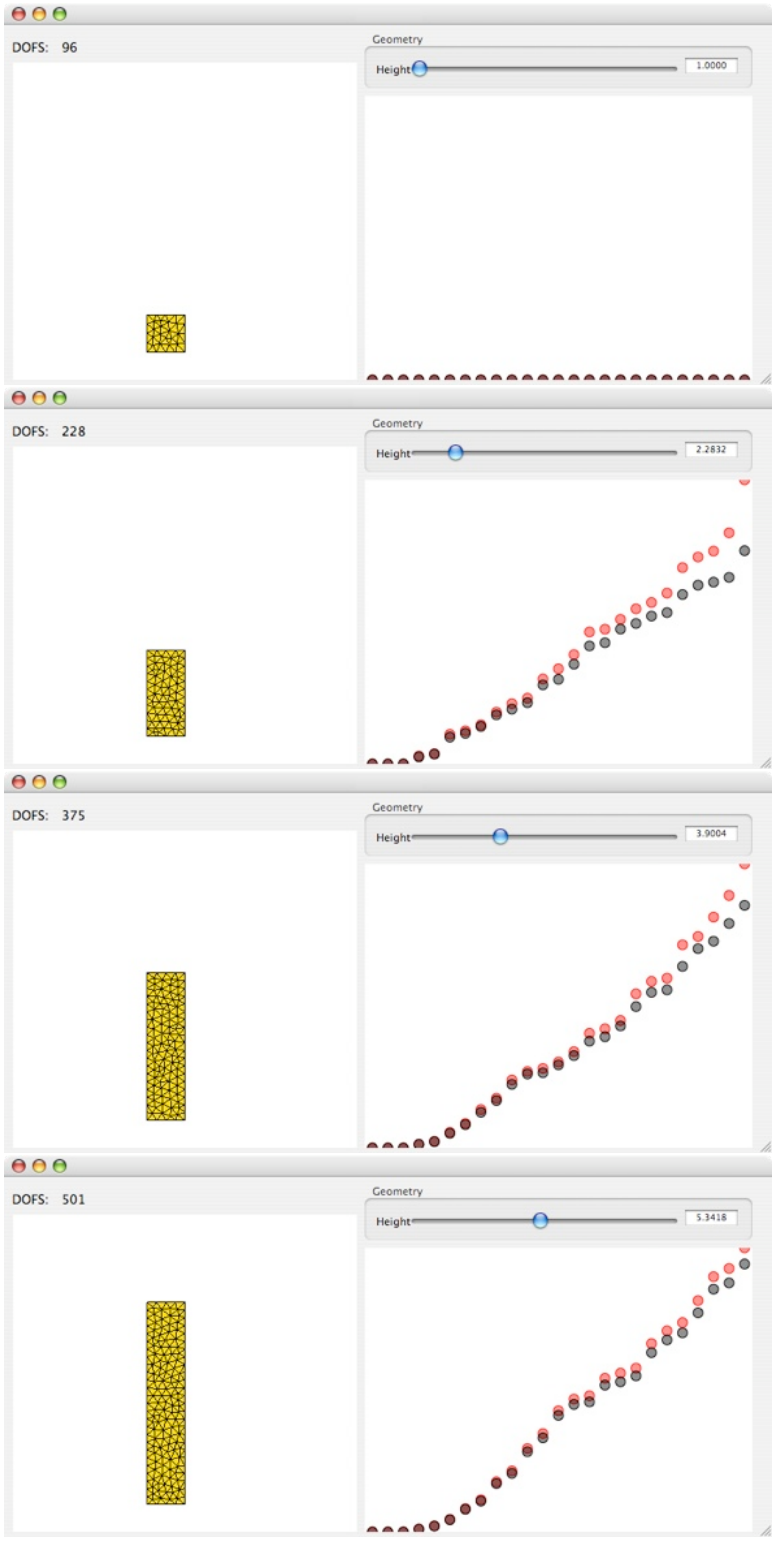

Figure 6.43. Four steps in an interactive shape-changing environment with re-meshing. Approximations to the spectrum (red dots) follow the true spectrum (black dots).

We know two methods to address the difficulty of completely resolving the invariant subspace corresponding to repeated eigenvalues. The first method is to use a block version of the standard Lanczos algorithm [66]. Unlike the ordinary Lanczos iteration we use, block Lanczos iteration can find an invariant subspace for an eigenvalue in a single step, provided the block size is the same or greater than the multiplicity of the eigenvalue. The second method is the radial decomposition technique described in Chapter 5, which uses analytical knowledge of the symmetry group that leads to the multiple eigenvalue in the first place.

Another source of problems is when a structure is much stiffer in some directions than in others. For example, our shell structure is much less resistant to out-of-plane bending than to in-plane compression. A vector that represents pure bending motion for one geometry may represent a mixture of bending and in-plane compression in a nearby geometry, so that a Rayleigh-Ritz approximation based on that vector will substantially overestimate the frequency at the new geometry.

Our results show that in practice, these problems do not cause significant errors in our prediction method.

6.5 Conclusions

The aim of this investigation is to determine if our tracking method can be used to predict the changes in the frequency spectrum of an object as parametric changes are made to its shape. The results of these experiments show that without remapping, it is possible to avoid recomputing the eigendecompositions in order to resolve the resonant frequencies of interest for moderate changes only. However, with geometric remapping, one can make significant changes to the geometry and still accurately predict the new frequency spectrum.

Even in the worst case when the mesh is significantly different, we can still accurately and rapidly predict the new spectrum.

By exploiting the properties of the system matrices, we can approximate the errors produced using different parameter step sizes. For an interactive design tool, this would mean that the software could alert the user when errors above a given threshold have been made and signal the need for a full reanalysis. This can be used in the tuning stages of design.

For systems with many repeated eigenvalues, such as axisymmetric systems, it may be beneficial to use analysis techniques that will factor out the multiple eigenvalue problem.

This investigation demonstrates that for interactive design applications, it is advantageous to track the spectrum for changes in geometry to avoid computing a partial eigendecomposition. By using this method, we can maintain a moderate linear time algorithm with increasing system size.

Using this rapid evaluation procedure, we can now create a software instrument that will allow one to hear the resulting frequency spectrum as changes are made to an object's shape.

Chapter 7

Software Instrument

Our system presents a novel use of 3D models for audio synthesis. Previous research into sound generation using modal analysis of complex 3D models of objects rendered the sound of objects colliding in an off-line computation [63]. Our goal was to generate sound in real-time allowing a user to feel as though she is "playing" the object by applying forces to a physical interface. Therefore, we incorporated the sound synthesis routines into a digital synthesizing plug-in that takes as input 3D geometric data. By implementing the system as a software synthesizer, we can interact with the object using software hosts that support this plug-in. Using this design allows for integration with music interfaces such as a piano keyboard.

7.1 Audio Units

For interactive sound generation, this software has been written as an Audio Unit [6]. Audio Units are Apple Incorporated's audio plug-in standard utilizing the Core Audio framework [8]. These plug-ins are hosted within performance and sequencing environments

allowing musicians to chain instruments and effects together, creating a virtual studio. As a result, this synthesizer can be used in any host application that supports Audio Unit plug-ins. Examples of such hosts are software packages such as Garage Band [9], Logic [10] and RAX [37].

The design of this plug-in can be decomposed into the synthesis algorithms used, the design of the user-interface and the overall architecture of the performance environment. The synthesis algorithms used have been described in Chapter 2.2. Here we describe the user-interface and the system architecture.

7.2 Software System Architecture

The software is written in C++ [1] and Objective-C [11], and uses the Cocoa [7] and OpenGL [2] APIs for the user-interface. The audio engine for the plug-in utilizes the Core Audio and [8] Altivec [5] APIs. The 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. Figure 7.1 is a diagram of the audio system.

7.3 Static Geometric Models

The system described in Bruyns [20, 21] reads in static geometry and pre-computed eigen-information. In this way the software synthesizer simply reads in the resonance information and presents a parametric audio rendering system. This system then, would be similar to any other system that rendered resonance models, say from an Sound Design Interchange Format (SDIF) file [86].

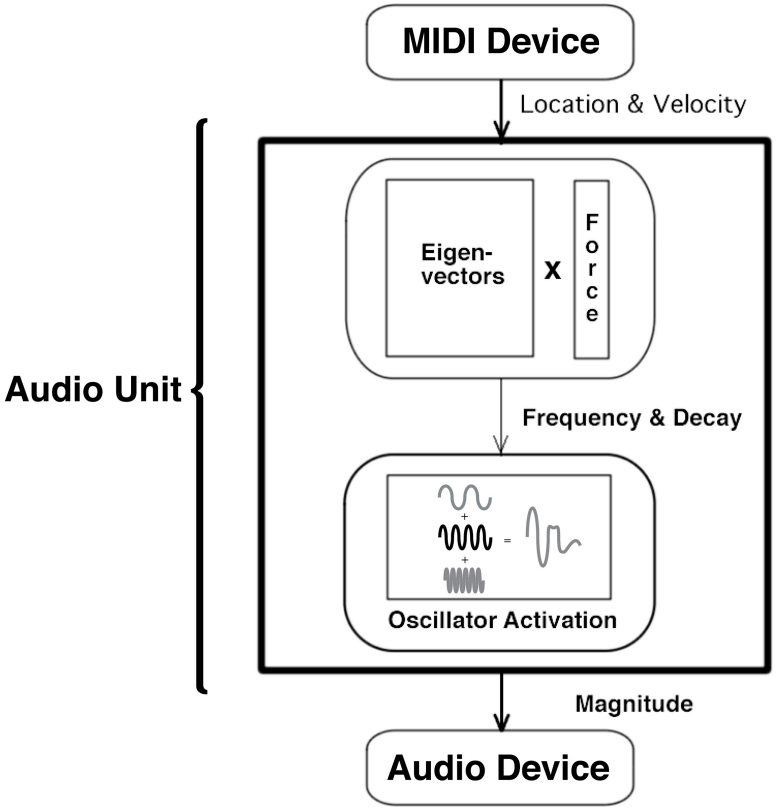

Figure 7.1. Sound synthesis engine.

The system we describe in this section also uses geometric models but instead computes eigen-information inside of the plug-in. This allows for modification to the material and shape parameters of the object. Each new material and mesh resolution requires updating the eigen-information using techniques described in Chapter 6.

For many software synthesizers, the user-interface (UI) provides an abstraction of the underlying synthesis model. Typically, in these systems resonating components are not modeled directly but are instead represented symbolically by their parameters.

Our plug-in, however provides visual feedback showing the changes to the parameters and the geometry used in the synthesis computations. Figure 7.2 shows the UI for the Virtual Instrument Audio Unit. The top portion contains the sliders for parameter adjustment, and the bottom portion gives a 3D view of the model to define the strike position and to examine the mode shapes.

The parameters that the user can control, corresponding to the sliders at the top of Figure 7.2 are:

1. Size of object
2. Material
3. Damping parameters α_1 and α_2
4. Resolution of the mesh
5. Number of modes used for the computation
6. Radius of the striking object
7. Base impulse applied to the object (that MIDI key-press velocity then scales)

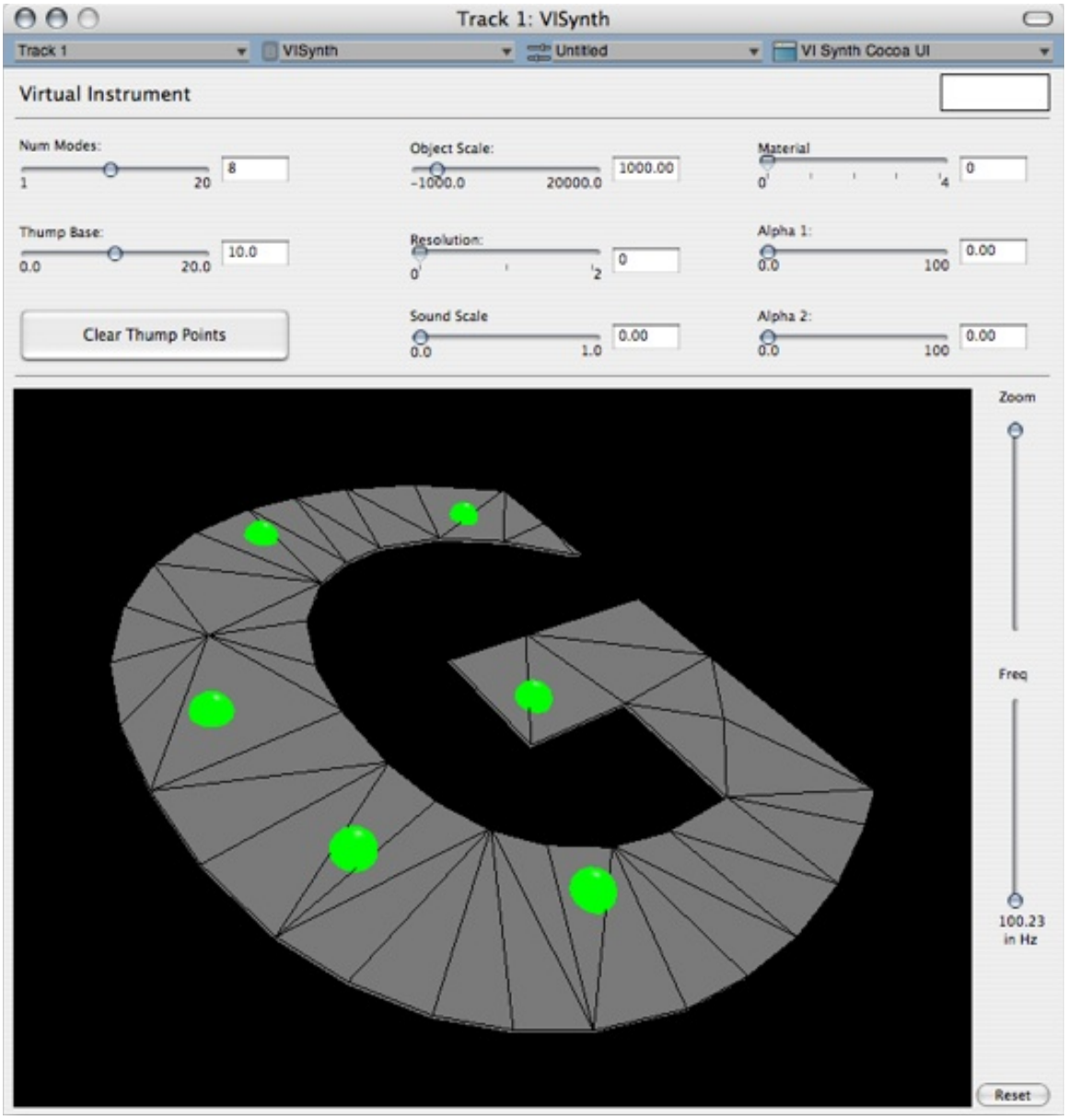

Figure 7.2. Audio Unit UI.

8. Volume control

Using the frequency scale to lower or raise the natural frequencies affects the perceived object size and material. Alternatively, one can adjust the material to achieve the desired spectral profile. One can also examine the results of the modal decomposition by iterating through the mode shapes. A slider selects the mode vibrating at a natural frequency (whose value is displayed at the bottom of the slider) and displays the corresponding shape deformations in the viewing window.

Using the mouse, the user selects a few specific locations to strike the object. The radius of the striking object determines the area over which the force is applied, i.e. force applied at a single location versus force applied over a larger surface area. The force is applied by examining which nodes are in the striking area and then applying force to them as in Equation 2.8.

The striking 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 strike direction is determined by the angle between the viewing direction and the normal of the surface at the strike location. Figure 7.3 shows the system in use.

7.4 Dynamic Geometric Models

We use the fast mode calculation technique described in Chapter 5 in a synthesizing plug-in for playing rotationally symmetric percussive instruments undergoing shape change. The geometry of the instrument is parameterized, and the positions of the control points that determine the geometry are mapped to controllers of a MIDI device. In this way, we

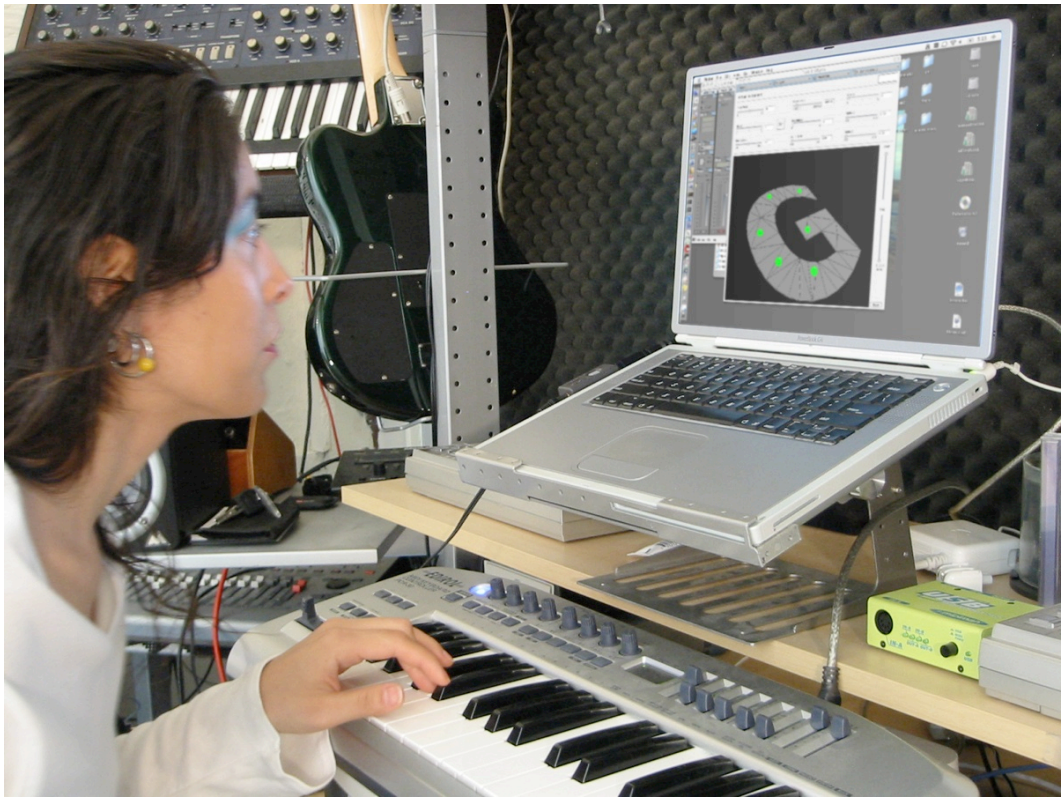

Figure 7.3. User playing the virtual instrument.

can modify the geometry using the same controllers we use to play the model. The result of this method is a system that allows a user to hear the result of modifying an axisymmetric object geometry interactively by allowing the user to strike the object while the shape is changing.

Figure 7.4 shows the user interface for the Shape Changing Audio Unit. The top portion of the user interface contains the sliders for parameter adjustment, and the bottom portion gives a three-dimensional view of the actual geometric model. The model is used to define the strike position and examine the mode shapes.

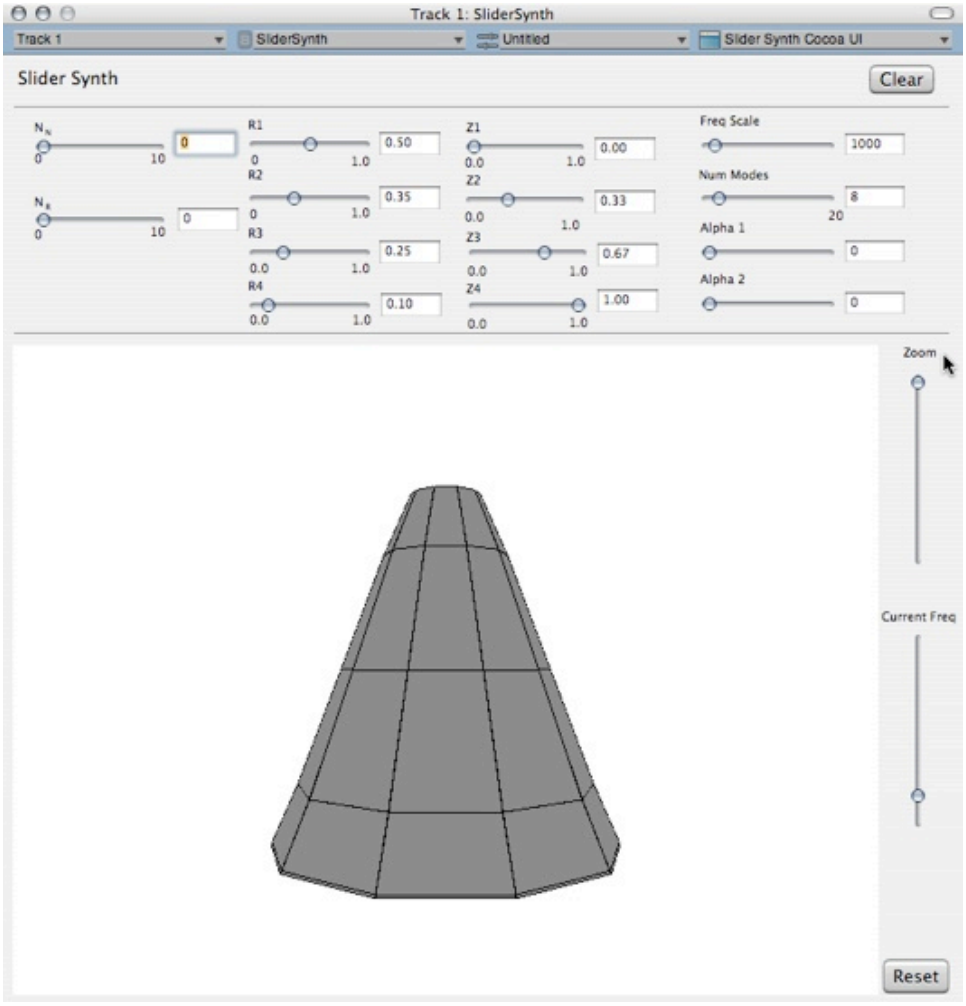

Figure 7.4. Synthesizer user interface.

The parameters that the user can control, corresponding to the sliders at the top of the plug-in, are as follows: the material parameters such as damping (α_1 and α_2); audio rendering parameters such as number of resonators (Num Modes) used, and a frequency scaling (Freq Scale); geometric parameters such as number of radial (N_R) and lateral (N_N) segments, as well as height (Z), and radii (R) of the control points.

Again, the user interacts with the object by 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.

Using this technique, we can generate sounds from objects as geometric modifications are made and then hear the changes in timbre as a function of shape. Figure 7.5 shows four different coarse models made from modifying the control points. By varying the radii of the control points, we change the shape of the resulting geometry. Figure 7.6 shows that as the radii of the different segments are changed, the peaks in the spectrum move in ways that would otherwise be difficult to predict. While the peaks stay within the 200 to 2000 Hz range, the strength[1] and number of the peaks vary for each shape. This is interesting given that the material of the object remains constant. Recall that the strength of a mode depends on its activation for a given impulse. Even though the location of the applied impulse remains constant for each of the four shapes, each shape will have a different response to this same strike. By examining Figure 7.7 the linear sonogram of the tones produced from each shape, we can again see the shift in sustained partials as the radii are changed. These results verify that changes in an object's geometry will move

[1]The variation in strength should be observed on the macro-level as these tones were generated live via MIDI keyboard. That is, because the applied force can vary from one key-strike to the next, one can generate different applied forces causing the peaks to be higher or lower.

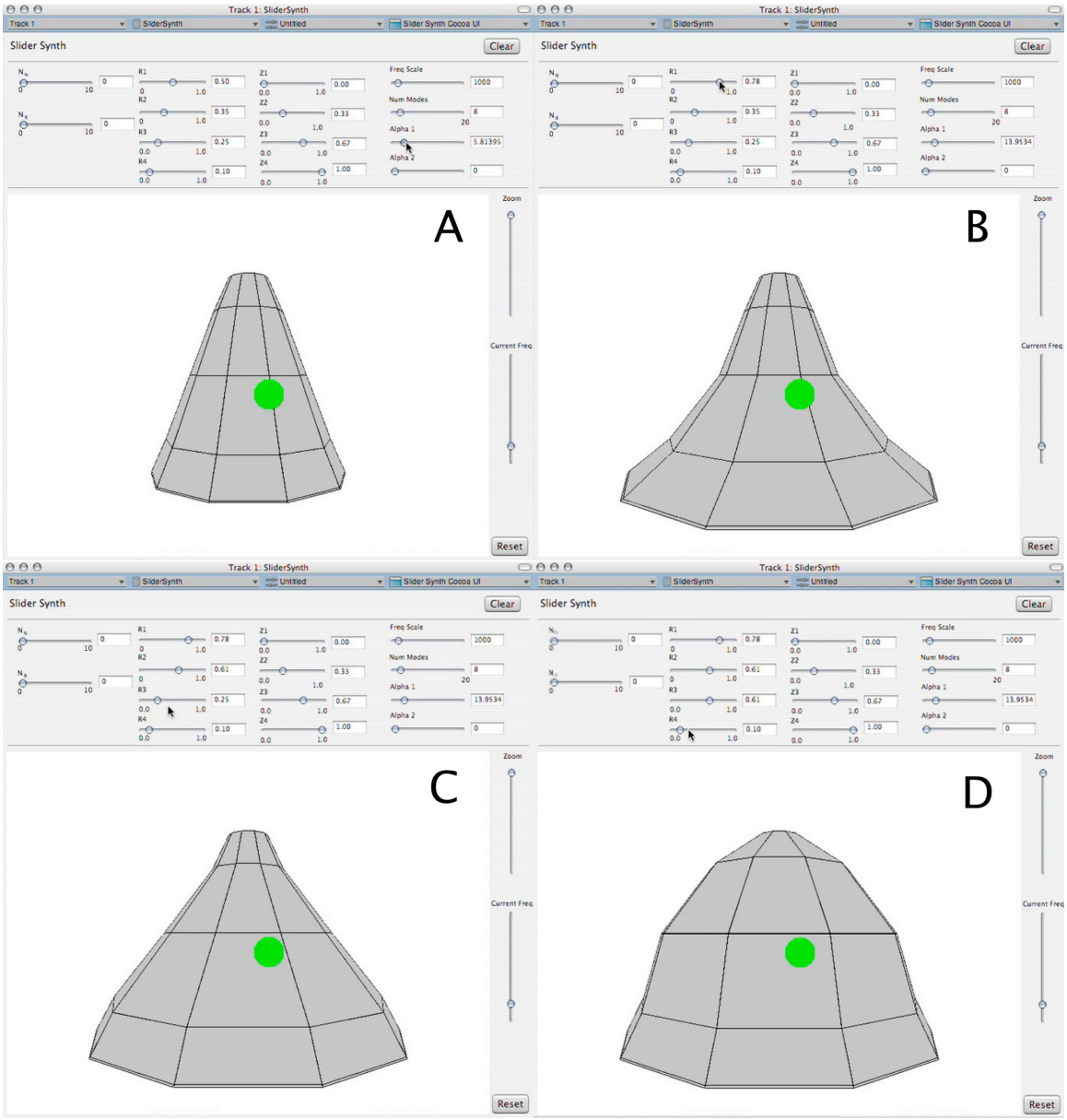

Figure 7.5. Deformed parametric shapes producing different frequency spectra.

peaks in its frequency spectrum. These changes are driven by the changes in the resonant frequencies and resonant shapes for different objects. By using fast decomposition methods for rotationally symmetric geometries, one can quickly explore how the peaks move and begin to investigate visual and aural aesthetic combinations for instrument design. These

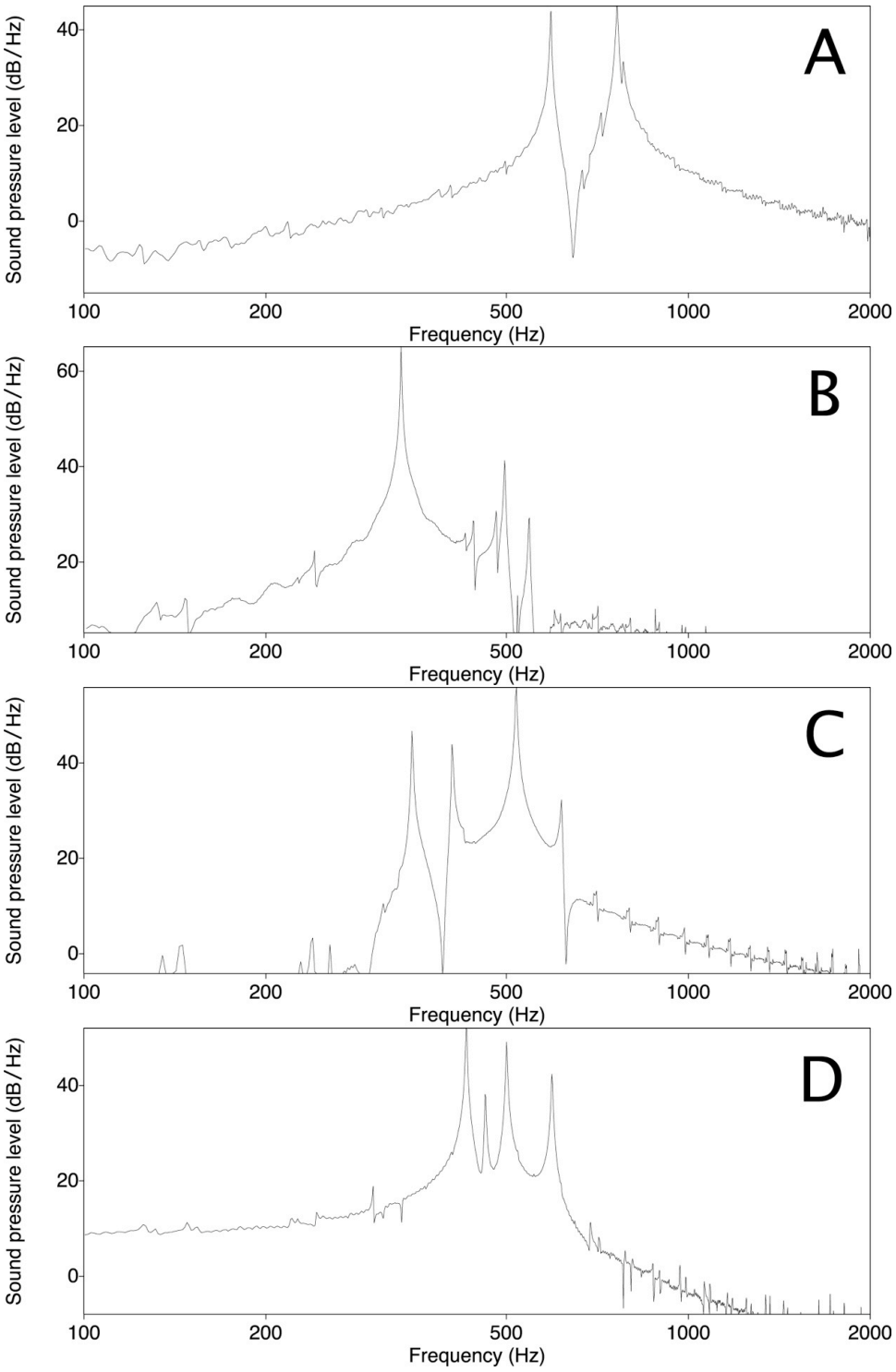

Figure 7.6. Frequency spectrum comparison corresponding to the four deformed shapes.

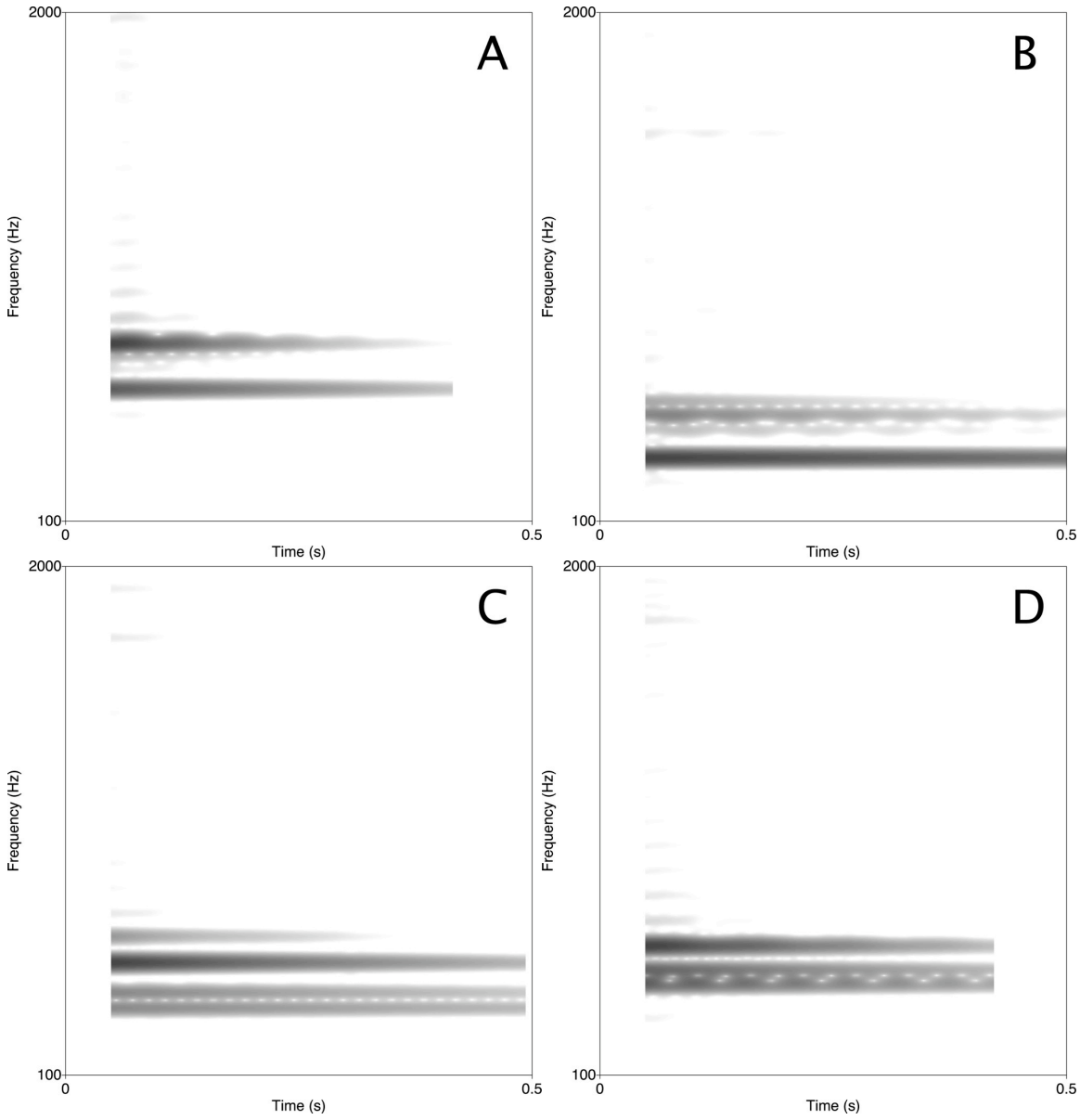

Figure 7.7. Sustained resonant frequencies of the four different shapes.

results also highlight the subtle interplay between shape and material that will determine the overall timbre of such objects.

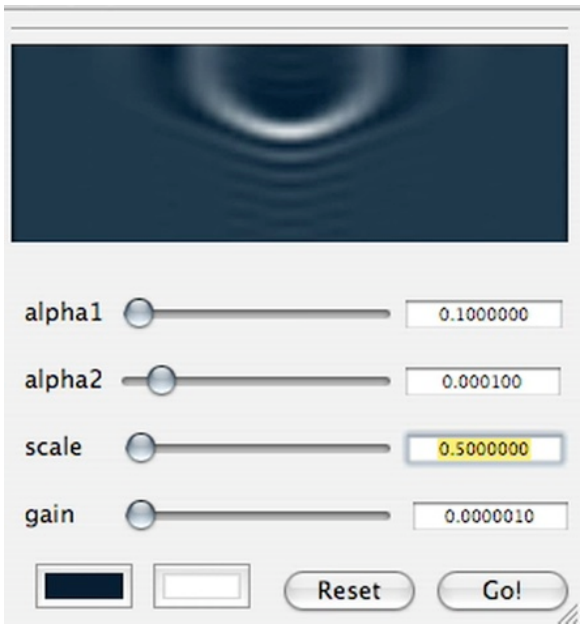

Figure 7.8. User interface for the fluid-coupled synthesis system.

7.5 Acoustic-Coupled Instrument Model

For the fluid-coupled object vibration simulations described in Bruyns [22] we allow the user to control the damping applied to the system, the size of the geometry, and the gain on the output. These controls can create very different radiation profiles and resulting tones. Figure 7.8 shows these controls and the region of the user interface where instead of rendering a 3D geometry, we render a 2D area representing the surrounding fluid. Using this system, we can generate in real-time an animation of the radiation profile and the sound generated by the object.

Because the sound is rendered at 44.1 kHz, if we were to run the radiation animation at the same sample rate, the pressure wave would equalize too rapidly to get a sense for its propagation. So instead, we compute the visual representation at 60 Hz to focus in on the initial radiation propagation after impact.

This system uses Apple Incorporated's Core Image API to render the pressure distribution to an OpenGL view. For each point in the fluid grid, we solve for the pressure distribution and compute an average pressure for the fluid element. We then create an image that is the size of the overall fluid area and run a fragment program on the image for image smoothing. The result is an efficient, plausible image of the propagating pressure wave.

Example sounds produced from both of these systems are available at the author's website:
http://cynthia.code404.com/thesis/.

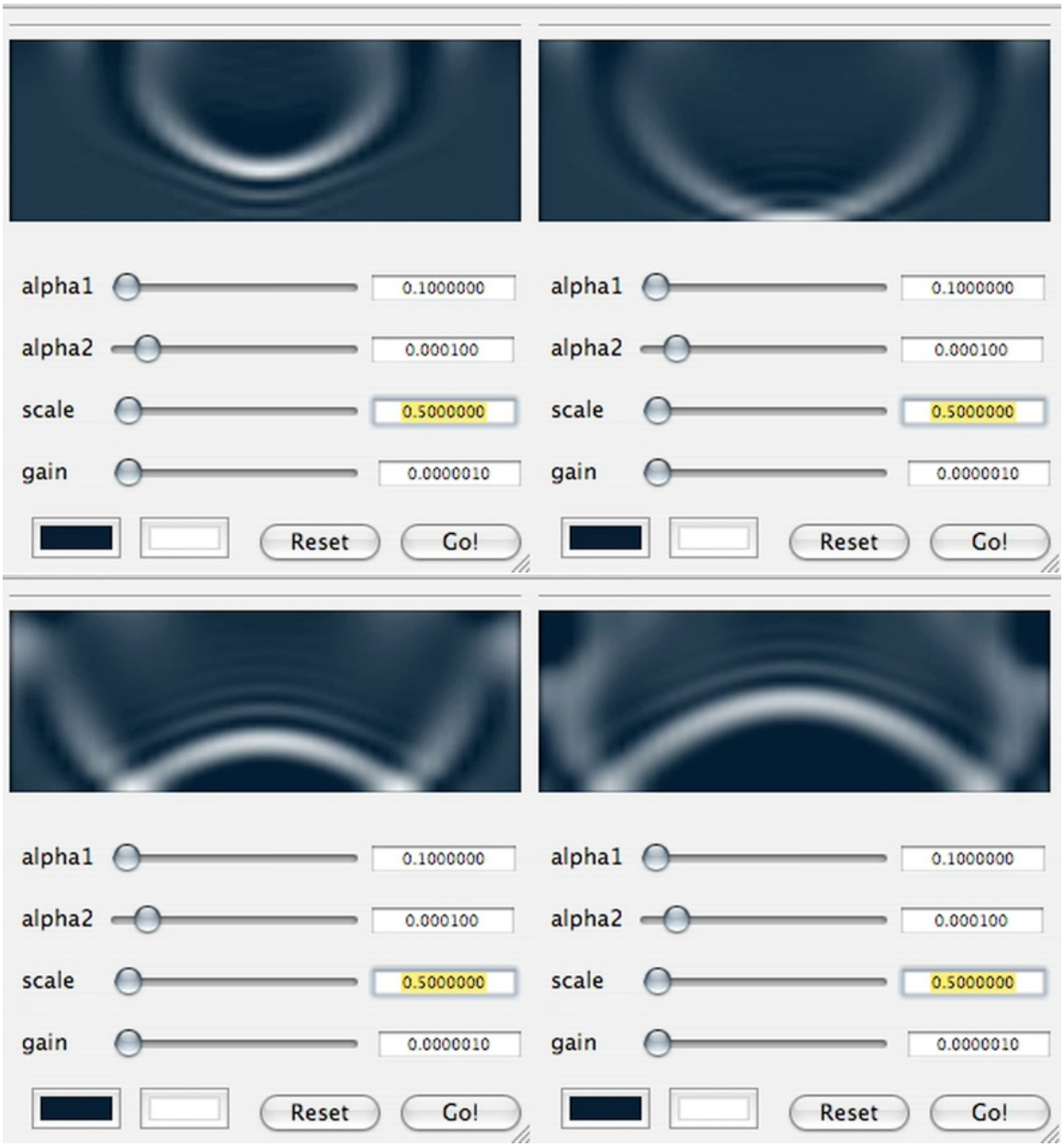

Figure 7.9. Propagating pressure wave rendered in real-time inside of a virtual instrument.

Chapter 8

Software Effect

Once modal decomposition is performed and the model is broken up into uncoupled resonators, one can interact with the model quite efficiently to generate sound. Chapter 2.2 examined methods for computing the response of a single degree-of-freedom (SDOF) oscillator to arbitrary input. In this chapter we examine ways of generating complex force profiles and show how to use these profiles to generate sound.

The basic system for generating sounds from geometric models was discussed in Chapter 3. As described previously, the sound synthesis engine is programed as a virtual instrument plug-in which receives MIDI data to start the audio rendering process. To support musical gestures that are more complex than a single strike (or impulse) to the object, however, this system was modified to support controllers that send more complicated force profiles.

In this chapter we describe methods of generating arbitrary force profiles from controller data, and we describe a modification to the original instrument system to utilize it as an audio effect.

8.1 Common Control Systems

The system presented in Chapter 7 maintained a process that listened for an incoming MIDI signal. This signal then alerted the audio processing engine to value of a given MIDI controller (such as keys on a keyboard, position of sliders, angles of a modulation wheel, etc).

For this system, the key controllers were mapped to locations on the surface of an object so that when their value changed, the force applied to the surface also changed. Because many MIDI devices have velocity-sensitive controllers, one could simulate striking the object with varying force by applying varying levels of velocity to these controllers.

Other controllers can be mapped directly to the synthesis parameters, allowing for flexible and smooth modification of the synthesized sound.

The engine listening for MIDI signals also tracks the current state of the controller, such as attack (when initial contact with the surface is made), sustain (when object is allowed to vibrate), and release (when vibration is damped out). One can use these different states to add more realism to the rendering engine, such as adding transients on attack to simulate bouncing and friction. For interactions that have longer contact times, one can use the sustain state to indicate that micro-contact (such as scraping or rolling) is occurring.

In the previous implementation, only the attack and release states were used, as we assumed a simple Dirac delta or impulse loading as in Equation 2.40.

There are controllers which use natural gestures to generate force profile data. For example, a haptic feedback device as used in DiFilippo and Pai [31] can be used to link the sound synthesis engine with user-perceived applied forces. Other controllers, such as the

drum pads, wind controllers, and special-purpose voltage-to-MIDI conversion devices can be used to generate these complex force profiles.

8.2 Complex Control Systems

There are sound rendering environments already in use that can leverage the resonance models created by our system. By exporting our models into a format, such as the Sound Design Interchange Format (SDIF) [86], our models can be used in several other rendering systems such as AudioSculpt [19], Loris [35], CHANT [82], SPEAR [48], and the very popular Max/MSP [28].

SDIF is a flexible file format for representing audio signals in a number of ways. In our case, we would utilize the representation for exponentially decaying sinusoids, or resonance, models. In this way, the result of our analysis would be a group of sinusoids decaying at a rate determined by the material. The SDIF file then is simply a list of properties for rendering the group of sinusoids, i.e. frequency, amplitude, decay rate, and optionally initial phase. In this format, the resulting synthesis is still flexible enough to allow for parametric modification. This is in contrast to simply generating a summed waveform, which would lose its parametric control.

There are also systems that support the SDIF format and multi-touch controllers. These controllers are extremely sensitive, providing continuous, polyphonic, reactive, and temporally precise synchronization with audio [84]. By linking our system with these new controllers, we expect musically interesting sounds to be generated from our models.

As we demonstrate in the next section, one can also use an arbitrary waveform, instead of

simply controller data, as input to our model. In this way, one can use the bank of resonators (which are the result of modal decomposition) to simulate artificial reverberation.

8.3 Simulated Transducer Loading

Historically, plate reverberation was used as a synthetic means to simulate large room acoustics. It was one of the first types of artificial reverberation used in recording [62]. Despite the unnatural sound produced as compared to large room reverberation, plates were used extensively due to their relative low cost and small size. Recently, researchers have looked for a means of digitally simulating plate reverberation to recreate this unique analog sound treatment [18].

Analog plate reverberation works by mounting a large thin steel plate under tension, which is supplied by springs at the corners where the plate is attached to a stable frame. The tension is used to keep the thin plate rigid and to control resonance. A signal from a transducer is applied to the plate, causing it to vibrate. This vibration is then sensed elsewhere on the plate with contact microphones. A nearby absorbing pad can also be used to control the near-field radiation and add in additional acoustic damping.

Before the prevalence of physical modeling, plate reverberation was simulated by recording impulse responses of plates made from different materials and different geometries. By convolution of the input with the impulse response of choice, the resulting audio could sound as though it was recorded through an actual plate. Although this method is very efficient, it is limited by the number of, and variations in, the recordings available.

Bilbao et al. [18] demonstrated a model for plate reverberation using a linear Kirchoff plate formulation. By using a physical model instead of convolution with impulse responses,

they could modify the geometry of the plate and input/output parameters. To simulate plate vibration, the model was discretized in space and time using finite differences. One drawback of this method, however, was the high performance required to run their model, which prevented them from running in real-time on an average digital audio workstation.

Using the modal synthesis method, we can compute a plate reverberation model in real-time and still allow for modifications of the plate and input/output parameters. To achieve this performance, we use the same finite element model as described in Bruyns [21] and apply forces using the discrete convolution integral method as described in DiFilippo and Pai [31] and van den Doel et. al. [79]. We implement this reverberation as an effect plug-in that takes an audio stream as the input and produces the sound of the object vibration as the output.

The main contributions of this chapter are to extend the use of the modal synthesis and the discrete convolution integral for the rapid simulation of the motion of objects in response to arbitrary loading profiles. We give examples of using this technique for the deformation of linear shell models of simple and complex shapes in a real-time synthesis environment.

We use the shell model as descried in Kwon and Bang [49], and perform modal analysis as described in Chapter 2.1 to arrive at the uncoupled bank of resonators. To solve for the motion at the pickup locations we weight the contributions of the various modes on the motion at spatial positions of interest.

To apply the input to the system, we use the recursive discrete convolution integral as described in Chapter 2.2. In this way, the incoming audio signal represents an arbitrary discretized force profile applied to the resonators interpolating the input position.

8.4 Software System Architecture

The rendering algorithm works by first performing the modal decomposition and then filtering the incoming audio with the resonator bank produced. The time to compute the modal decomposition depends on the number of modes required and the number of elements in the finite element model. We achieve real-time performance by first computing the decomposition, which can take several seconds[1]. We then evaluate Equation 2.51 for each audio sample.

The UI for the plug-in loads an object geometry and displays the surface for specifying the input and pickup locations. The left portion of the UI allows for modification of the material parameters, object scale and plate thickness. These parameters are adjusted before modal decomposition. The right portion of the UI has controls for the audio rendering parameters such as the frequency scaling and resonator decay. These parameters do not require reanalysis; instead they are applied to the bank of resonators as audio is rendered. One can also control the number of resonators used for simulation. Using more resonators creates a fuller tone, but requires more computation.

8.5 Results

The following examples were computed using one processor of a dual 2.5GHz PowerPC G5. The plug-ins were hosted using Apple Incorporated's AULab application and audio input was streamed using the built-in AUFilePlayer component. In each example (Figure 8.2),

[1]We only compute the decomposition at the start of the audio rendering. In Chapter 6 we describe techniques for rapidly computing this decomposition when the object is changing shape and other changes are made to the model.

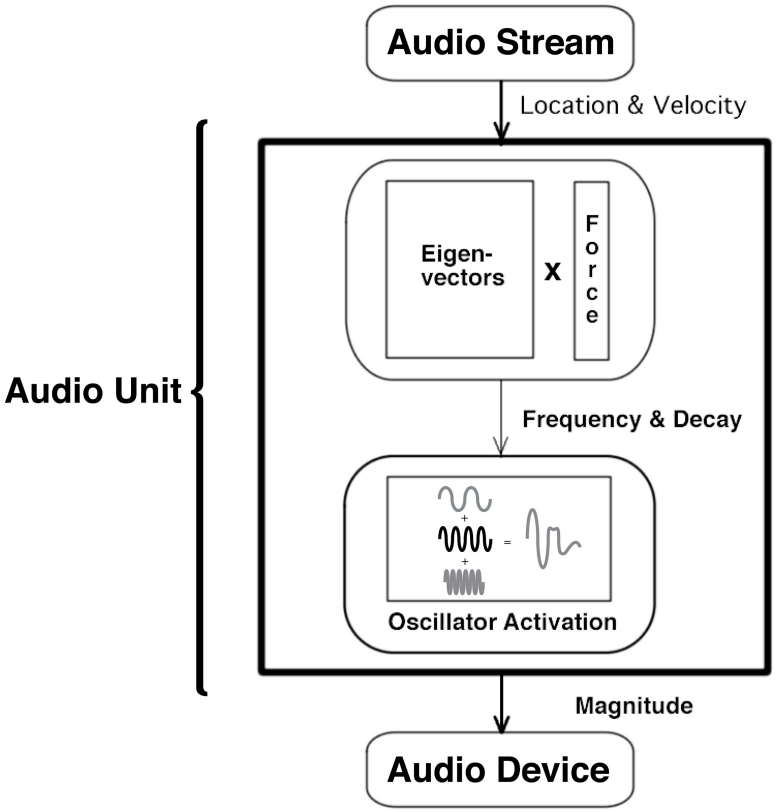

Figure 8.1. Reverb as an effect on a stream of audio.

the points in green represent the input position and the points in red represent the pickup locations.

For the first example, we load a simple plate model as shown in Figure 8.2 (top). The model has 100 elements, and the time to compute the decomposition into 485 modes was 0.65 seconds. Figure 8.3 (top) shows the waveform, and Figure 8.4 (top) shows the spectrogram of the incoming signal applied to the plate. Figure 8.3 (middle) shows the resulting waveform, and Figure 8.4 (middle) shows the frequency profile generated for the left channel. Immediately, one can see the effect of reverberation on the resulting audio. Where there were once brief waveform impulses, the audio is now stretched-out in time and blends

together to form broadened pulses. Moreover, the frequency spectrum is low-pass filtered through the number of modes used in the synthesis algorithm.

We can also use this method on novel shapes and explore the effect on the resulting audio. Figure 8.2 (bottom) shows a more complex shell surface with arbitrary input and output locations. This model had 500 elements and it took 24.5 seconds to compute all 1548 modes. Using the same input profile as Figure 8.3 (top), we can compare the resulting waveform and frequency spectra when rendering with this new geometry (Figure 8.3 (bottom), Figure 8.4 (bottom)).

In Figure 8.4, the output of the resonator bank has fewer high frequency components than the original signal. This is to be expected, as user-selected damping values will alter the original signal. In some sense, the original signal acts to imprint its frequency spectrum on the resonator bank roughly, but the two spectra will not exactly match. In other words, if the plate was infinitely stiff, the applied impulse would exactly match the resulting signal, but as damping in the plate causes delay in the propagation of the sound wave, the spectra is shifted downwards.

For both of these examples, simulating object vibration using 20 modes consumed around 1.4% of the overall CPU capacity; 100 modes consumed roughly 3%; 1000 modes consumed 22%; and 3000 modes used 84% for two channels of stereo processing. These results suggest that for up to 1000 modes, the method performs well. For the 3000 or so resonators needed for non-metalic, perceptually-realistic sounding reverberation [45], the real-time CPU demand is considerable when using only one processor. In practice, one rarely needs that many resonators to be synthesized simultaneously.

8.6 Discussion

For plates with very thin cross-sections, it is likely that large applied forces will cause large plate deformation. When this happens, linear models can no longer be used. As a result, techniques such as linear modal superposition must be abandoned for nonlinear modal analysis or nonlinear models and numerical integration. Other researchers are actively investigating the importance of these nonlinearities in plate reverberation models [23].

Many times, plate reverberation systems are used with a parallel absorbing plate to control the radiated sound. In these cases, air is trapped in between the two plates, changing the resonant behavior of the system. In Bruyns [22], we examine the effect of sound radiation and fluid-structure coupling on the timbre of an object.

In this investigation we have demonstrated a technique for simulating reverberation using the modal synthesis method. By using a physical model of a vibrating object, we are free to use any arbitrary geometry and material.

This method has the advantage over simulated analog modeling because it allows the user to explicitly place the transducer and pickup locations along the surface of the object. Moreover, this method combined with shape-changing anlaysis, creates a novel reverberation technique that can simulate reverberation from a shape-changing room.

Example sounds produced from this system are available at the author's website: http://cynthia.code404.com/thesis/.

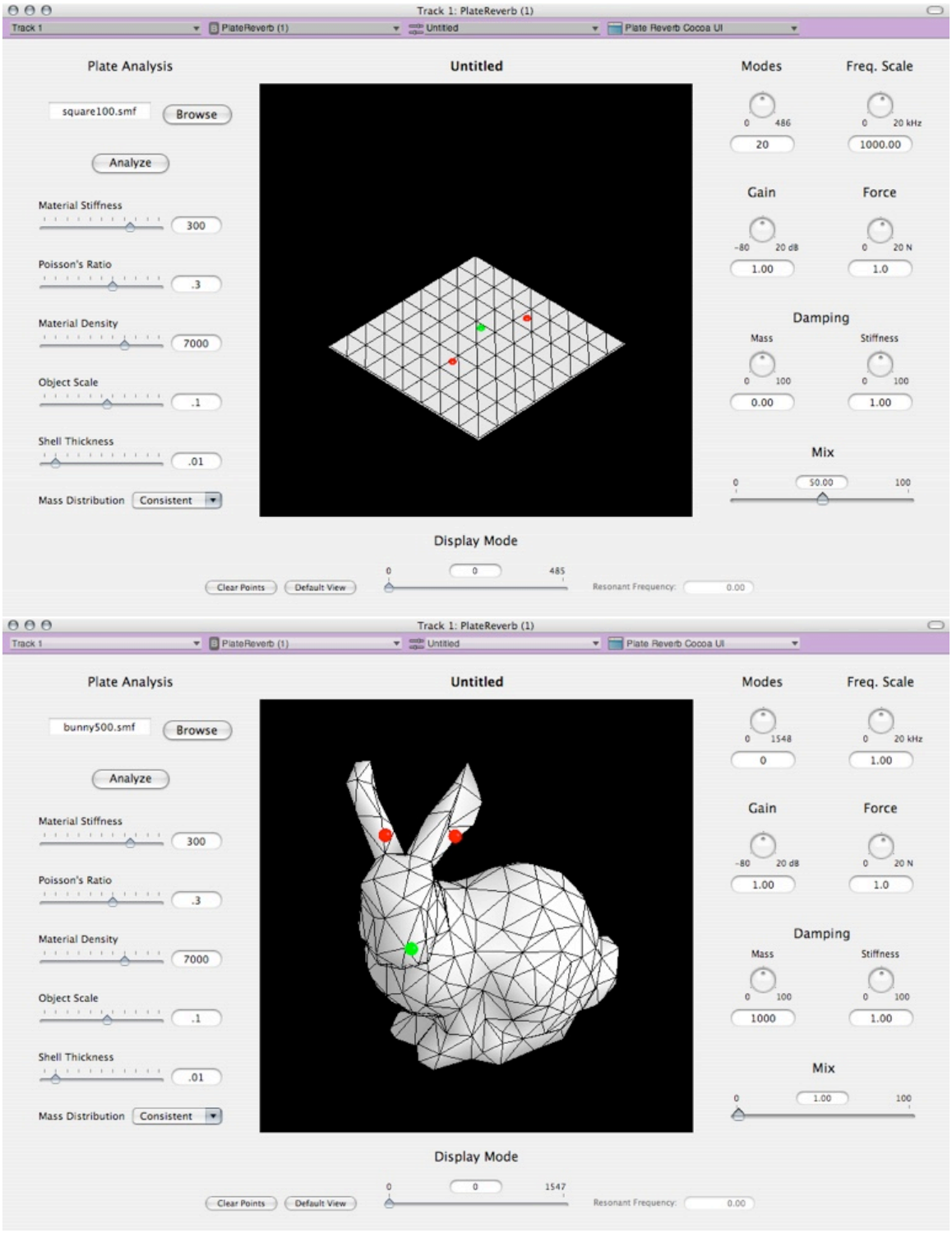

Figure 8.2. Top: A traditional plate reverb geometry. Bottom: A complex bunny shaped reverb surface.

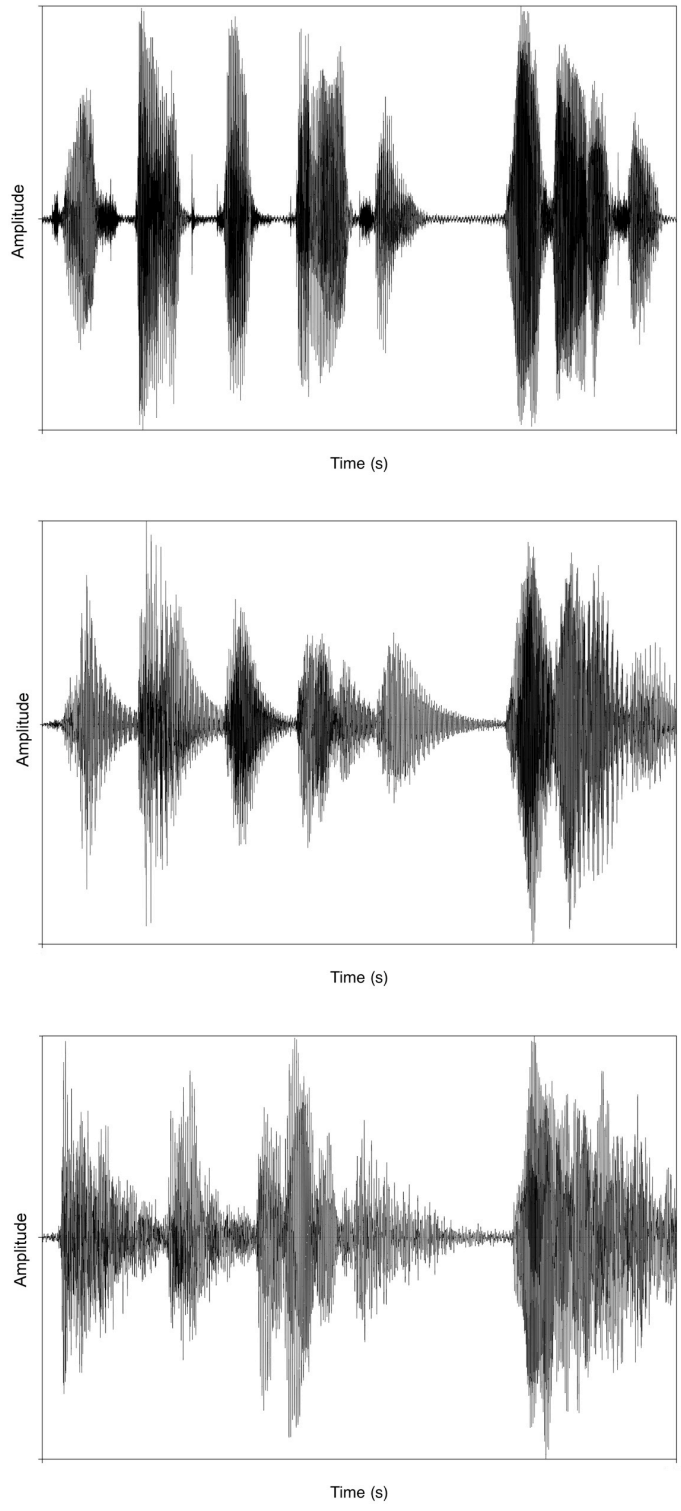

Figure 8.3. Top: Force profile applied to the plate. Middle: Waveform output from simple plate vibration. Bottom: Waveform output from complex bunny shape surface vibration.

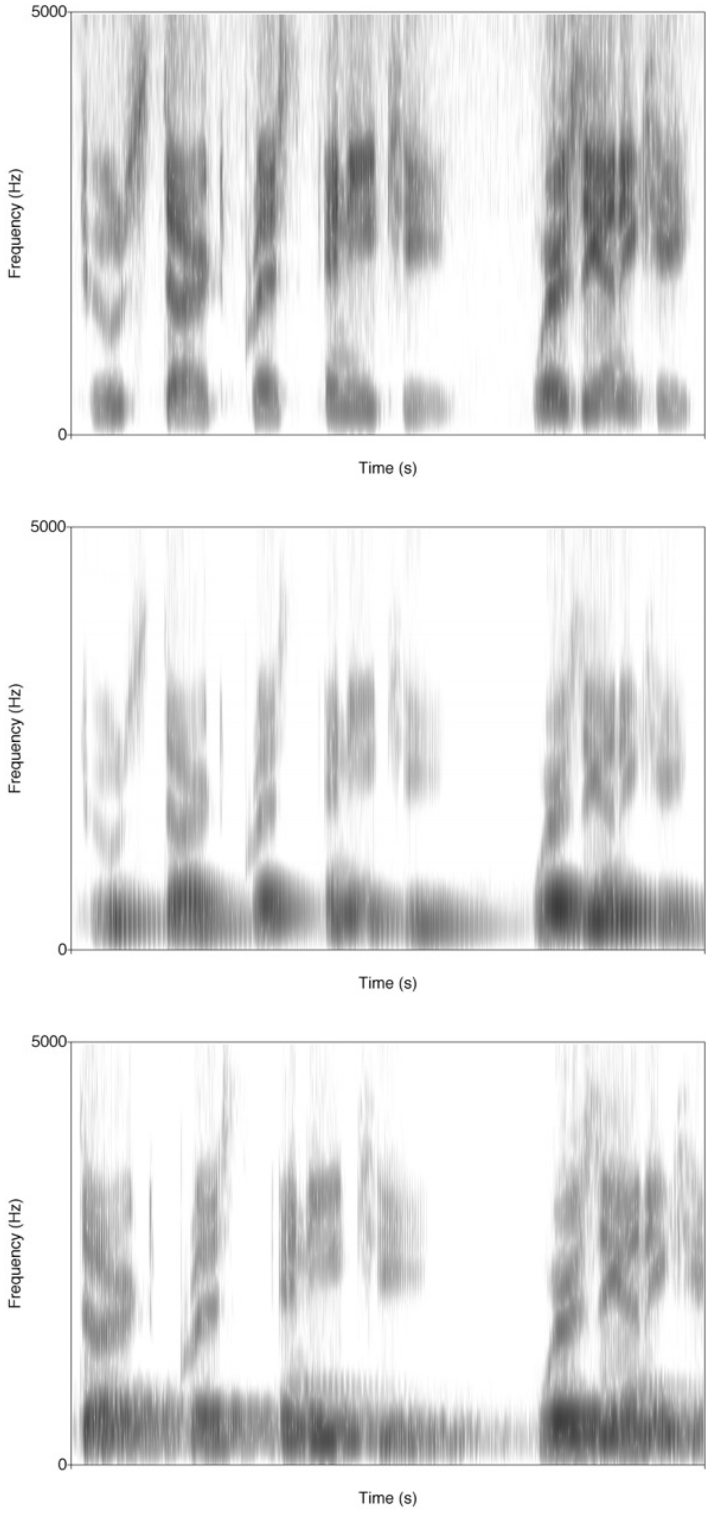

Figure 8.4. Top: Frequency spectrum of the input force profile. Middle: Spectral output from simple plate vibration. Bottom: Spectral output from complex bunny shape surface vibration.

Chapter 9

Conclusions

This research extends real-time, physics-based sound synthesis in computer music. Using our system, one can synthesize sounds from three-dimensional models of objects in an interactive music setting.

Using modal synthesis, the mechanical vibration of an instrument can be represented efficiently once a full or partial eigenvalue decomposition of the system matrices is computed. This eigenvalue problem is relatively expensive. However, one only needs to compute the decomposition once for a given instrument. The goal of this research is to design instruments in real-time. That is, we would like to know precisely how changing the shape or the material of an instrument will change its timbre. Because the timbre of an object depends strongly on its shape, designing a new instrument shape with standard modal analysis tools would require a new decomposition for each new design – a prohibitively expensive step for an interactive tool.

In this dissertation, we described ways of rapidly estimating the modal parameters of an object as the object's shape is changing. The method exploits properties of parameter-dependent linear systems by tracking an invariant subspace as modifications are made.

Using this method, one can avoid recomputing the spectrum while still providing an accurate representation of an object's timbre. The results show very high accuracy for moderate changes and a significant speed increase when computing the modal parameters required for sound synthesis.

These new techniques provide an alternative framework for instrument design and create a platform for interacting with instruments. Our system also allows for creative exploration of the design of instruments that could not exist in the real world today, such as those made from metals that can change shape without application of heat or force. Or instruments made from materials that would normally break under the application of striking forces, but in this system they vibrate according to their specific material properties. One can envision adding such melting and breaking physics to generate even more interesting sounds.

We implemented these techniques in a real-time software interface to demonstrate the efficiency, and compelling interactivity, of this new mode of sound synthesis.

This method has wide-ranging applications, such as the design of musical instruments, loudspeaker casings, and architectural spaces.

Bibliography

[1] A history of C++. http://portal.acm.org/citation.cfm?id=155375.

[2] OpenGL. http://www.opengl.org/.

[3] S. Adhikari. Damping modeling and identification using generalized proportional damping. In Proceedings of the 23rd International Modal Analysis Conference (IMAC-XXIII), Orlando, Florida, USA, Feb 2005.

[4] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. D. Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. LAPACK Users' Guide. Third edition, 1999.

[5] Apple Incorporated. Altivec. http://developer.apple.com/hardware/ve/index.html.

[6] Apple Incorporated. Audio Units. http://developer.apple.com/audio/audiounits.html.

[7] Apple Incorporated. Cocoa. http://developer.apple.com/cocoa/.

[8] Apple Incorporated. Core Audio. http://developer.apple.com/audio/coreaudio.html.

[9] Apple Incorporated. Garage Band. http://www.apple.com/ilife/garageband/.

[10] Apple Incorporated. Logic Pro. http://www.apple.com/logic/.

[11] Apple Incorporated. The Objective-C programming language. http://developer.apple.com/documentation/Cocoa/Conceptual/ObjectiveC/.

[12] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst. Templates for the Solution of Algebraic Eigenvalue Problems: A PracticalGuide. SIAM, 2000.

[13] J. Barbič and D. L. James. Real-time subspace integration for St. Venant-Kirchhoff deformable models. ACM Transactions on Graphics (SIGGRAPH 2005), 24(3):982–990, Aug. 2005.

[14] K. J. Bathe and E. L. Wilson. Numerical Methods in Finite Element Analysis. Prentice Hall, 1976.

[15] Z. P. Bazant. Spurious reflection of elastic waves in nonuniform finite element grids. Computer Methods in Applied Mechanics and Engineering, 16:91–100, 1978.

[16] T. Belytschko and R. Mullen. Modern Problems in Elastic Wave Propagation, chapter On dispersive properties of finite element solutions, pages 67–82. International Union of Theoretical and Applied Mechanics. Wiley, 1978.

[17] M. A. Bhatti. Advanced Topics in Finite Element Analysis of Structures: With Mathematica and MATLAB Computations. Wiley, 2006.

[18] A. Bilbao, K. Arcas, and A. Chaigne. A physical model for plate reverberation. IEEE International Conference on Acoustics, Speech and Signal Processing, 5:V–165–V–168, 2006.

[19] N. Bogaards, A. Röbel, and X. Rodet. AudioSculpt. http://forumnet.ircam.fr/691.html.

[20] C. Bruyns. Virtual instrument design and animation. Master's thesis, University of California at Berkeley, 2005.

[21] C. Bruyns. Modal synthesis for arbitrarily shaped objects. Computer Music Journal, 30(3):22–37, 2006.

[22] C. Bruyns-Maxwell. A simple simulation of acoustic radiation from a vibrating object. Proceedings of AES 123, 2007.

[23] A. Chaigne. Plate reverberator: modeling, analysis and synthesis. CCRMA Music 420 Seminar, March 2007.

[24] E. Chicurel-Uziel. Closed-form solution for response of linear systems subjected to periodic non-harmonic excitation. In Proceedings of the Institute of Mechanical Engineers, Part K, 2000.

[25] A. K. Chopra. Dynamics of Structures Theory and Applications to Earthquake Engineering. Prentice Hall, 2000.

[26] P. Cook. Music, Cognition, and Computerized Sound. MIT Press, 2001.

[27] R. R. Craig and A. J. Kurdila. Fundamentals of Structural Dynamics, chapter 10, pages 298–300. Wiley, 2006.

[28] Cycling74. Max/MSP. http://www.cycling74.com/products/maxmsp.

[29] P. Davis. Circulant Matrices. John Wiley & Sons Inc., 1979.

[30] J. W. Demmel. Applied Numerical Linear Algebra. SIAM, 1997.

[31] D. DiFilippo and D. Pai. The AHI: An audio haptic interface for contact interactions. In Proceedings of ACM UIST 2000, 2000.

[32] P. Djoharian. Material design in physical modeling sound synthesis. Journal of New Music Research, 30(3):227–241, 2001.

[33] F. Fahy. Sound and Structural Vibration: Radiation, Transmission and Response. Academic Press, 1987.

[34] S. Felszeghy. On uncoupling and solving the equations of motion of vibrating linear discrete systems. Journal of Applied Mechanics, 60(2):456–462, 1993.

[35] K. Fitz and L. Haken. Loris. http://www.cerlsoundgroup.org/Loris/.

[36] N. H. Fletcher. The nonlinear physics of musical instruments. Reports on Progress in Physics, 62:723–764, 1999.

[37] Granted Software. RAX. http://www.grantedsw.com/rax/.

[38] J. He and Z.-F. Fu. Modal Analysis. Butterworth-Heinemann, 2001.

[39] N. Holmes and T. Belytscho. Postprocessing of finite element transient response calculationsby digital filters. Computers & Structures, 6:211–216, 1976.

[40] A. Horner, J. Beauchamp, and R. So. Detection of random alterations to time-varying musical instrument spectra. Journal of the Acoustical Society of America, 116(3):1800–1810, 2004.

[41] J. H. Hwang and F. Ma. On the approximate solution of nonclassically damped linear systems. Journal of Applied Mechanics, 60:695–701, 1993.

[42] L. Jiang and R. R. Rogers. Effects of spatial discretization on dispersion and spurious oscillations in elastic wave propagation. International Journal for Numerical Methods in Engineering, 29:1205–1218, 1990.

[43] D. Jones. Viscoelastic Vibration Damping. Wiley, 2001. pages. 1-16, 215-252.

[44] R. Kapania and C. Byun. Reduction methods based on eigenvectors and Ritz vectors for nonlinear transient analysis. Computational Mechanics, 11:65–82, 1993.

[45] M. Karjalainen and H. Järveläinen. More about this reverberation science: Perceptually good late reverberation. In In Proceedings of the AES 111th International Convention, September 2001.

[46] L. E. Kinsler, A. Frey, A. Coppens, and J. Sanders. Fundamentals of Acoustics. Wiley, 1999.

[47] R. Klatzky, D. Pai, and E. Krotkov. Perception of material from contact sounds. Presence: Teleoperators and Virtual Environments, pages 399–410, 2000.

[48] M. Klingbeil. SPEAR. http://www.klingbeil.com/spear/.

[49] Y. W. Kwon and H. Bang. The Finite Element Method Using MATLAB. CRC Press, 2000.

[50] S. Lakatos, S. McAdams, and R. Causse. The representation of auditory source characteristics: simple geometric form. Perception & Psychophysics, 59(8):1180–1190, 1997.

[51] R. Lakes. Viscoelastic Solids, chapter 4, page 126. CRC Press, 1998.

[52] A. Lehr. Partial groups in the bell sound. Journal of the Acoustical Society America, 79(6):2000–2011, 1986.

[53] A. Leissa. Vibration of Plates. NASA-SP-160, 1969.

[54] F. Ma and T. Caughey. Analysis of linear nonconservative vibrations. Journal of Applied Mechanics, 62:685–691, 1995.

[55] R. I. Mackie. Improving finite element predictions of modes of vibration. International Journal for Numerical Methods in Engineering, 33(2):333–344, 1992.

[56] M. E. McIntyre, R. T. Schumacher, and J. Woodhouse. On the oscillations of musical instruments. Journal of the Acoustical Society of America, 74(5):1325–1345, 1983.

[57] L. Meirovitch. Principles and Techniques of Vibrations. Prentice Hall, Upper Saddle River, New Jersey, first edition, 1999.

[58] L. Meirovitch. Fundamentals of Vibration. McGraw Hill, 2001.

[59] J. Meriam and L. Kraige. Engineering Mechanics: Dynamics, volume 2, chapter 8, pages 595–600. Wiley, 3rd edition, 1992.

[60] I. A. Murra and G. M. L. Gladwell. On a search for a major-third and other non-standard bells. Journal of Sound and Vibration, 149(2):330–340, 1991.

[61] I. A. Murra and G. M. L. Gladwell. On the effect of curvature on the nodal circles of bell modes. Journal of Sound and Vibration, 150(2):317–321, 1991.

[62] D. C. Myers. Audio reverberator. Technical report, U.S. Patent 4,653,101, March 24, 1984.

[63] J. F. O'Brien, C. Shen, and C. M. Gatchalian. Synthesizing sounds from rigid body simulations. In Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. New York: Association for Computing Machinery, pages 175–181, 2002.

[64] I. Ojalvo and M. Newman. Vibration modes of large structures by an automatic matrix-reduction method. AIAA Journal, 8(7):1234–1239, 1970.

[65] M. M. Ozakca and M. T. Gogus. Structural analysis and optimization of bells using finite elements. Journal of New Music Research, 33(1):61–69, 2004.

[66] B. Parlett. The Symmetric Eigenvalue Problem (Classics in Applied Mathematics). SIAM, Philadelphia, 1998.

[67] R. Perrin and T. Charnley. Group theory and the bell. Journal of Sound and Vibration, 31(4):411–418, 1973.

[68] R. Perrin and T. Charnley. A comparative study of the normal modes of various modern bells. Journal of Sound and Vibration, 17(3):411–420, 1987.

[69] R. Perrin, T. Charnley, and J. DePont. Normal modes of the modern English church bell. Journal of Sound and Vibration, 90(1):24–49, 1983.

[70] S. S. Rao. Vibration of Continuous Systems, chapter 17, pages 661–670. Wiley, 2007.

[71] M. Rath. Some things that you can do with an impact sound. In Proceedings of DS2002, March 20-21 2002. Journees design sonore a Paris.

[72] M. Rath. An expressive real-time sound model of rolling. In In Proceedings DAFx-03, 2003.

[73] T. D. Rossing and R. Perrin. Vibration of bells. Applied Acoustics, 20:41–70, 1987.

[74] T. D. Rossing, R. Perrin, H. J. Sathoff, and R. W. Peterson. Vibrational modes of a tuned handbell. Journal of the Acoustical Society America, 76(4):1263–1276, 1984.

[75] A. Shapiro. Remap: A computer code that transfers node information between dissimilar grids. Technical Report UCRL-ID-104090, Lawrence Livermore National Lab., CA (USA), Apr 1990.

[76] G. Strang and G. Fix. An Analysis of the Finite Element Method. Prentice-Hall, 1973.

[77] P. Tong, T. H. H. Pian, and L. L. Bucciarelli. Mode shapes and frequencies by finite element method using consistent and lumped masses. Computers & Structures, 1:623–638, 1971.

[78] K. van den Doel, D. Knott, and D. K. Pai. Interactive simulation of complex audiovisual scenes. Presence: Teleoperators and Virtual Environments, 13(1):99–111, 2004.

[79] K. van den Doel, P. G. Kry, and D. K. P. . Foley automatic: Physically-based sound effects for interactive simulation and animation. In Proceedings of the 2001 ACM SIGGRAPH Conference, pages 537–544, 2001.

[80] K. van den Doel, D. Pai, T. Adam, L. Kortchmar, and K. Pichora-Fuller. Measurements of perceptual quality of contact sound models. In Proceedings of the 2002 Conference on Auditory Display, 2002.

[81] E. W. van Heuven. Acoustical Measurements of Church-Bells and Carillons. S-Gravenhage De Gebroeders van Cleef, 1949.

[82] D. Virolle. CHANT. http://recherche.ircam.fr/equipes/analyse-synthese/DOCUMENTATIONS/libchant/index.html.

[83] G. B. Warburton and S. R. Soni. Errors in response calculations for non-classically damped structures. Earthquake Engineering & Structural Dynamics, 5(4):365–376, 1977.

[84] D. Wessel, R. Avizienis, A. Freed, and M. Wright. A force sensitive multi-touch array supporting multiple 2-d musical control structures. In NIME 2007, 2007.

[85] E. L. Wilson and J. Penzien. Evaluation of orthogonal damping matrices. International Journal for Numerical Methods in Engineering, 4(1):5–10, 1972.

[86] M. Wright, A. Chaudhary, A. Freed, D. Wessel, X. Rodet, D. Virolle, R. Woehrmann, and X. Serra. New applications of the sound description interchange format. In Proceedings of the 1998 Int. Computer Music Conf., pages 276–279, 1998.