

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Alternative Representations of the Two Dimensional Waveguide Mesh

Permalink

<https://escholarship.org/uc/item/0r17j2jw>

Author

Done, Ryan Michael

Publication Date

2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA, SAN DIEGO

Alternative Representations of the Two Dimensional Waveguide Mesh

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Arts

in

Music

by

Ryan Done

Committee in charge:

Professor Tamara Smyth, Chair
Professor Miller Puckette
Professor Shahrokh Yadegari

2017

Copyright
Ryan Done, 2017
All rights reserved.

The thesis of Ryan Done is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2017

TABLE OF CONTENTS

Signature Page	iii
Table of Contents	iv
List of Figures	vi
List of Tables	vii
Acknowledgements	viii
Abstract of the Thesis	ix
Chapter 1 Introduction	1
1.1 Introduction	1
1.1.1 Motivation	1
1.1.2 Goals	2
1.1.3 Complementary Work	2
1.2 Organization	3
1.2.1 Filter Orders	3
1.2.2 Performance Timings	3
Chapter 2 The Two Dimensional Waveguide Mesh	5
2.1 2D Digital Waveguide Mesh Physical Model	5
2.1.1 Boundary Conditions	7
2.1.2 Excitation and Output	7
2.2 Difference Equations	8
2.2.1 Inputs and Outputs	9
2.3 Comparison to Other Methods	10
2.3.1 Mesh Analysis and Alternative Representations	11
2.4 Implementation and Performance	12
Chapter 3 The Mesh as a Transfer Function	14
3.1 Mesh Symmetry	14
3.1.1 Difference Equation	15
3.1.2 Performance Timings	15
3.1.3 Constrained Input/Output	15
3.2 Transfer Function Analysis	16
3.2.1 Symbolic Computation	17
3.2.2 Center Excitation	17
3.2.3 Square Mesh with Corner Excitation	18
3.3 Usage of Symbolic Transfer Function	19
3.3.1 Coefficient Quantization	20

	3.3.2 Second Order Sections	22
Chapter 4	State Space Representation of Mesh	25
	4.1 State Space Model	25
	4.2 State Space Analysis	27
	4.2.1 Insights about Mesh Transfer Function	28
	4.3 Analysis in a Fixed Precision Environment	28
	4.3.1 Implementation Results	29
	4.4 Parallel Second Order Sections	32
Chapter 5	Approximation of Mesh Parameters	33
	5.1 Effects of Varying the Reflection Coefficient	33
	5.1.1 Choosing an Approximation Function	33
	5.1.2 Second Order Sections	35
	5.2 Trajectory of Zeros and Poles	37
	5.2.1 Polynomial Approximation	37
	5.2.2 Zero Pole Tracking	38
	5.3 Approximating Excitation Location	39
Chapter 6	Conclusion	43
	6.1 Transfer Function of the Mesh	43
	6.2 Parametric Instrument Design	43
	6.2.1 Boundary Filters	44
	6.3 Modal Representation	44
	6.4 Complexity	45
Appendix A	Waveguide Difference Equations for Border Cases	46
Bibliography		49

LIST OF FIGURES

Figure 2.1:	3x3 mesh of scattering junctions and delay elements	6
Figure 2.2:	Sinusoidal excitation shape at center of 50x50 mesh. Times $n = 0, 10, 20$. Code used from [1].	9
Figure 3.1:	Normalized spectra of 3x3 mesh for different excitations	16
Figure 3.2:	Zero pole plots from truncated symbolic transfer functions, $\mathbf{a} = 1$	20
Figure 3.3:	Zero pole plots with truncated zeros and poles, $\mathbf{a} = 1$	21
Figure 5.1:	Pole zero plots of 3x3 for range of $\mathbf{a} = [0.5, 1]$	34
Figure 5.2:	Time domain plots of 3x3 for range of $\mathbf{a} = [0.5, 1]$, note the decreasing time scale of the x-axis (time)	35
Figure 5.3:	Normalized spectra of 3x3 mesh for actual and $M + 1$ th root approximated boundary for range of $\mathbf{a} = [0.5, 1]$	36
Figure 5.4:	First quadrant trajectories of zeros and poles for 3x3 and 5x5 over $a = [0.5, 1]$. Other quadrants are symmetric.	37
Figure 5.5:	3x3 angle and radius curves for the first quadrant. Outer angle corresponds to angle of zero or pole at $\mathbf{a} = 1$	38
Figure 5.6:	Normalized spectra of 3x3 mesh for actual and polynomial ap- proximated boundary for range of $\mathbf{a} = [0.5, 1]$	39
Figure 5.7:	Magnified quadrant trajectories of zeros and poles where crossover occurs for 5x5 and 7x7 over $a = [0.5, 1]$. Other quadrants are symmetric.	40
Figure 5.8:	Zero pole plot of parallel second order sections, colors represent vertically paired complex conjugate poles and zero that form a section.	41
Figure 5.9:	Frequency response of various excitations points in the symmet- ric upper left quadrant of 7x7 mesh. Outline specifies unique possible excitations.	42
Figure 5.10:	Frequency response actual (blue) and approximated (orange) excitations points in the symmetric upper left quadrant of 7x7 mesh. Outline specifies unique possible excitations.	42

LIST OF TABLES

Table 3.1:	Benchmarking of difference equations for mesh with center excitation	16
Table 3.2:	Results of Mathematica transfer function analysis	19
Table 3.3:	Results of C++ implementation of series second order sections . .	23
Table 4.1:	State space analysis and C++ timings - corner excitation	30
Table 4.2:	Center and edge excitation: zero pole order and SOS count . . .	31
Table 4.3:	Orders for every possible input output pair on the 3x3 mesh . .	31

ACKNOWLEDGEMENTS

A huge thank you to my committee members for taking the time to review this thesis, and to all my instructors and peers at UCSD who have built an enriching environment to learn and collaborate. The friends I have made at UCSD welcomed me with open arms when I arrived here two years ago, and have provided me with a wealth of support and encouragement.

Thank you to thank my parents, Ann and Brent Done for their incredible love, support, and for fostering a passion for music early on in my life. I would also like to thank Mrs. Cathey Tyler and Mr. Jason Iseli for investing in me so many years of wonderful music instruction.

Lastly, I would like to extend my utmost gratitude to my committee chair Professor Tamara Smyth for her invaluable guidance and mentorship, and for first introducing me to Computer Music with her course CMPT 468: Computer Music Theory and Sound Synthesis in the Fall of 2013 at SFU. Since programming my first synthesizer in her class, the field has captivated my interest.

This work was supported by a research grant from the Humanities Division of University of California San Diego, offering graduate student support.

ABSTRACT OF THE THESIS

Alternative Representations of the Two Dimensional Waveguide Mesh

by

Ryan Done

Master of Arts in Music

University of California, San Diego, 2017

Professor Tamara Smyth, Chair

The two dimensional waveguide mesh is one possible discrete representation of the wave equation based on the theoretical framework of the one dimensional waveguide model. The digital waveguide model provides intuitive parametric control of physical properties, making it conducive to musical synthesis. However, interesting inharmonic and percussive timbre from the modeling of membranes and surfaces has often been off limits to real-time synthesis due to the computational expense of larger two dimensional mesh models.

This thesis covers the results of investigating methods to convert the two dimensional waveguide mesh into transfer function representations that can be computed faster in real-time, without compromising the parameters of control that the physical model provides. Two methods for analyzing the mesh and extracting

a transfer function are explored, one conducive to deriving an analytical representation of the mesh, the other a numerical representation. Issues concerning implementation of analysis and the using the transfer functions as filters are investigated. Finally, methods for approximating the parametric control of the physical model in the filter representations are introduced.

Chapter 1

Introduction

1.1 Introduction

This document presents a summary of the research efforts in the Winter and Spring quarters of 2017 that focused on deriving and exploring alternative representations of the two dimensional waveguide mesh in order to construct real-time computer music instruments based on mesh sizes that were previously too large to support real-time synthesis.

1.1.1 Motivation

The 2D waveguide mesh is capable of synthesizing a rich palette of inharmonic timbres useful for percussion and spring reverb type sounds. However, its implementation has had limited application as an instrument due in part to its large size and computational complexity, which limits the possibility for real-time sound synthesis of interesting (larger) meshes. Other discretization methods of the 2D wave equation suffer from this performance burden as well, and plenty of existing research has attempted to alleviate the problem through the using high performance computing techniques, such as GPUs and other parallel processing architecture. An alternative approach to the problem is to attempt to reduce the representation of the waveguide mesh model to one where less information is required to compute an perceptually equivalent result. Such a solution would offer

a performance improvement in a platform agnostic way, allowing for improved real-time synthesis to be implemented on mobile or embedded platforms, or as a plug-in.

1.1.2 Goals

The material in this thesis begins with an investigation into methods to convert the two dimensional (2D) waveguide mesh physical model into an analytical representation in the frequency domain. Such a representation allows for immediate implementation in the time domain as an infinite impulse response (IIR) filter, and for the ability to inspect the analytical solution to hopefully reveal novel observations about how the waveguide mesh behaves. It is also expected that the IIR filter implementation will allow for faster audio synthesis over the original physical model. Since the intent is to use the resultant filters as mesh-like instruments, it is also necessary to maintain or reincorporate the physical model's parameters in the filter representation; analysis that simply produces a static transfer function is not nearly as useful for the indented application.

As will be detailed in the coming chapters, directly deriving the canonical IIR filter representation will encounter limitations that require alternative methods that produce factored representations. The limitations of analysis will also hinder the ability to carry forward the physical models parameters directly into the analytic solution, thus strategies to approximate the behavior of the mesh with respect to changing parameters in the new representation are investigated as well.

1.1.3 Complementary Work

The work here can be read as a companion piece to [2] and other work being prepared by Smyth, Hsu and Done which covers methods for deriving analytical representations of the 2D mesh. Implementation issues and more detailed results of the mesh analysis that were not originally documented are elaborated upon here.

1.2 Organization

This thesis will begin in chapter 2 by describing the formulation of the 2D waveguide mesh physical model, establishing a context of prior research, comparing it to other discretization methods of the 2D wave equation, and finally discussing how it has been used in existing software implementations. In chapter 3, the outcome of implementing the analytical method to derive the symbolic canonical transfer function established in [2] is covered. The limitations of the resulting canonical transfer functions are detailed, and zero pole and second order section representations are employed as solutions to the limitations of implementing large order IIR filters. Next, chapter 4 outlines a method for extracting the transfer function or zeros and poles of the mesh based on a state space representation of the physical model, as well as how the state space can be used to better construct second order sections. Similar to the previous chapter, analysis and implementation results of this method are presented. Finally, in chapter 5, approximations of the physical model's parameters are re-introduced to the static filter system derived in chapter 4 by manipulating the zeros and poles (and equivalently the second order section coefficients) of the new condensed representation.

1.2.1 Filter Orders

Results of analysis are often presented in terms of an IIR filter order (order of numerator over order of denominator), or equivalently number of zeros and poles. The interpretation of these numbers is two-fold: they indicate the amount of spectral content (number of peaks, timbre) that a filter (or bank of filters) emphasizes, and they also imply the computational expense of implementing the filter (the amount of filter state in memory, the number of operations per sample). Intuitively, spectral content is directly coupled with computational expense.

1.2.2 Performance Timings

Timings in this thesis are reported both for analysis time as well as filter implementations in audio synthesis. Since analysis is intended to run offline (in

Matlab or Mathematica), timing is merely provided to suggest to general feasibility of running analysis for various mesh sizes on a common PC. Filters were implemented in C++ and both analysis and audio synthesis ran on a Core i5 3.40GHz system with 8GB of RAM. Performance was profiled by measuring the amount of time to synthesize one second of audio at a sample rate of 44100Hz, a reasonable standard for real-time audio. If the time-to-compute result is less than one second, then we are assured that the particular synthesis method can be computed in real-time. Once again, these timing values are not intended to be a rigorous benchmark nor a statement of computational complexity, rather they are presented as empirical evidence that the filter representations allow for real-time audio synthesis as intended.

Chapter 2

The Two Dimensional Waveguide Mesh

2.1 2D Digital Waveguide Mesh Physical Model

The two dimensional (2D) waveguide mesh was originally introduced in [3] by Van Duyne and Smith, providing a discrete representation of the 2D wave equation based on the theoretical structure of the one dimensional (1D) waveguide. Where the 1D waveguide is useful for the synthesis of string and cylindrical bore sounds (such as woodwinds), the 2D is relevant to the modeling of membranes and surfaces, particularly for the synthesis of inharmonic percussion sounds such as cymbals. Wave propagation is formulated as a grid of four port scattering junctions interconnected by bidirectional delay elements. Figure 2.1 provides a visual intuition for how the formulation of the 2D mesh fits together. Each scattering junction has four pairs of positive (incoming) and negative (outgoing) ports. Borrowing from the formulation in [2], the velocity at scattering junction $J_{i,j}$ at time n can be expressed as:¹:

$$J_{i,j}(n) = \frac{1}{2}(N_{i,j}^+(n) + S_{i,j}^+(n) + E_{i,j}^+(n) + W_{i,j}^+(n)) \quad (2.1)$$

¹The N, S, E, W notation established in [2] will be used throughout this paper. Other notation conventions exist, such as [3] and [4].

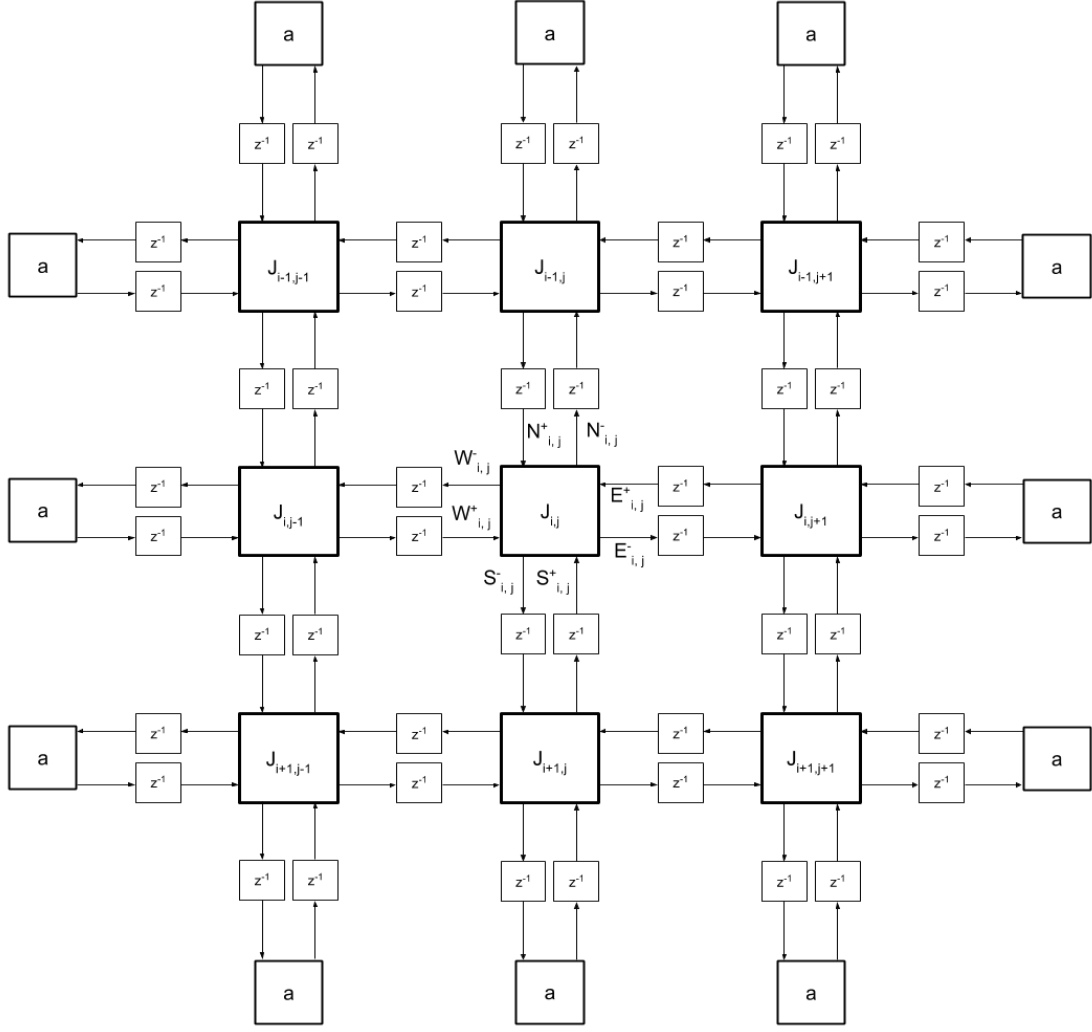


Figure 2.1: 3x3 mesh of scattering junctions and delay elements

The positive and negative traveling waves that enter and exit each scatter junction are expressed as follows:

$$\begin{aligned}
 N_{i,j}^-(n) &= J_{i,j}(n) - N_{i,j}^+(n) = \frac{1}{2}(-N_{i,j}^+(n) + S_{i,j}^+(n) + E_{i,j}^+(n) + W_{i,j}^+(n)) \\
 S_{i,j}^-(n) &= J_{i,j}(n) - S_{i,j}^+(n) = \frac{1}{2}(N_{i,j}^+(n) - S_{i,j}^+(n) + E_{i,j}^+(n) + W_{i,j}^+(n)) \\
 E_{i,j}^-(n) &= J_{i,j}(n) - E_{i,j}^+(n) = \frac{1}{2}(N_{i,j}^+(n) + S_{i,j}^+(n) - E_{i,j}^+(n) + W_{i,j}^+(n)) \\
 W_{i,j}^-(n) &= J_{i,j}(n) - W_{i,j}^+(n) = \frac{1}{2}(N_{i,j}^+(n) + S_{i,j}^+(n) + E_{i,j}^+(n) - W_{i,j}^+(n))
 \end{aligned}
 \tag{2.2}$$

$$\begin{aligned}
N_{i,j}^+(n) &= \begin{cases} aN_{i,j}^-(n-2), & \text{if } i = 1 (\text{north boundary}) \\ S_{i,j}^-(n-1), & \text{otherwise} \end{cases} \\
S_{i,j}^+(n) &= \begin{cases} aS_{i,j}^-(n-2), & \text{if } i = N (\text{south boundary}) \\ N_{i,j}^-(n-1), & \text{otherwise} \end{cases} \\
E_{i,j}^+(n) &= \begin{cases} aE_{i,j}^-(n-2), & \text{if } i = N (\text{east boundary}) \\ W_{i,j}^-(n-1), & \text{otherwise} \end{cases} \\
W_{i,j}^+(n) &= \begin{cases} aW_{i,j}^-(n-2), & \text{if } i = 1 (\text{west boundary}) \\ E_{i,j}^-(n-1), & \text{otherwise} \end{cases}
\end{aligned} \tag{2.3}$$

2.1.1 Boundary Conditions

As equation (2.3) denotes, the boundaries of a mesh are handled as special cases. When a wave exits a border scattering junction on a negative port (N^- for example), it passes through a two sample round trip delay, before returning to the same scattering junction through it's positive port. A parameter $\mathbf{a}$ where $-1 \leq \mathbf{a} \leq 1$ is used to model loss in the mesh at it's borders by scaling the returning wave along boundary edge. Intuitively, if $|\mathbf{a}| = 1$ then the mesh will have no loss of energy and will sustain indefinitely. As is consistent with the theory for digital waveguides, [3] explains how clamped boundaries that cause traveling waves to invert can be implemented using $\mathbf{a} < 0$ and unclamped boundaries can be implemented using $\mathbf{a} > 0$. For $|\mathbf{a}| < 1$, energy loss manifests itself as an exponential amplitude decay on the output. A scalar is not the only possible implementation of the boundaries in the mesh. Filters can be substituted for $\mathbf{a}$ on the boundaries, allowing for frequency dependent loss.

2.1.2 Excitation and Output

Much like the 1D case, output at a point in the mesh can be calculated by summing the incoming waves at a scattering junction: $y(n) = V_{i,j}(n)$. Multiple junctions can be summed together to form $y(n)$ as well. As it is pointed out in [3],

one of the properties of each scattering junction is that the sum of their outputs at time n is equal to the sum of their inputs at time $n - 1$. Because of this, it is not sufficient to excite a mesh by directly inputting to a junction $V_{i,j}(n)$, as doing so would cause a discontinuity in the output. An initial excitation must be applied to all the incoming ports of the junction that it being input to. Thus, to create a unit impulse at junction $V_{i,j}$ at time $n = 1$, all incoming ports must be set to $\frac{1}{2}$ in order to cause equation (2.1) to add up to 1:

$$N_{i,j}^+(0) = S_{i,j}^+(0) = E_{i,j}^+(0) = W_{i,j}^+(0) = \frac{1}{2} \quad (2.4)$$

It is possible to input a time varying signal $x(n)$ into the mesh by adding $\frac{1}{2}x(n)$ to the incoming port of each scattering junction.

$$\begin{aligned} N_{i,j}^+(n) &= N_{i,j}^+(n) + \frac{1}{2}x(n) \\ S_{i,j}^+(n) &= S_{i,j}^+(n) + \frac{1}{2}x(n) \\ E_{i,j}^+(n) &= E_{i,j}^+(n) + \frac{1}{2}x(n) \\ W_{i,j}^+(n) &= W_{i,j}^+(n) + \frac{1}{2}x(n) \end{aligned} \quad (2.5)$$

A wider variety of excitations may be achieved by exciting an area of junctions in the same manner according to some continuous function in 2D space. Figure 2.2 provides an example of a sinusoidal shaped excitation on a surface plot of the scattering junctions ($V_{i,j}$) for a 50 by 50 mesh. Perceptually, larger area excitations like the one pictured excite higher frequencies less.

2.2 Difference Equations

The formulation of the waveguide mesh can be rearranged into a system of difference equations. Equations (2.2) and (2.3) can be combined to eliminate the positive (+) junction port variables, leaving the negative (−) superscript unneces-

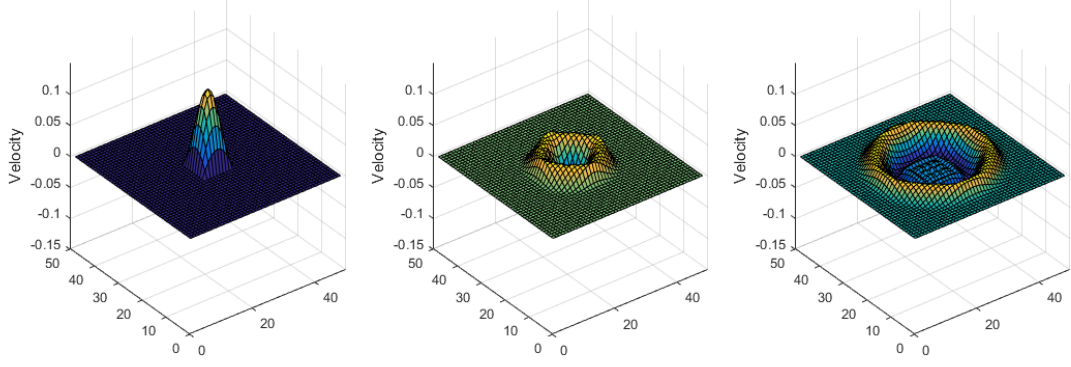


Figure 2.2: Sinusoidal excitation shape at center of 50x50 mesh. Times $n = 0, 10, 20$. Code used from [1].

sary. The N, S, E, W ports of a non-boundary junction can be expressed as:

$$\begin{aligned}
 N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\
 S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\
 E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) - W_{i,j+1}(n-1)) \\
 W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) - E_{i,j-1}(n-1) + W_{i,j+1}(n-1))
 \end{aligned} \tag{2.6}$$

Boundary junctions are similar to equation (2.6), but with the rules from 2.3 used where appropriate. Their specific formulations are available in appendix A. In this scheme, the $V_{i,j}$ variables need not be maintained. The inputs and outputs for a specific scattering junction $V_{i,j}$ can simply be calculated instantaneously using the sum of junction ports associated with the output position.

2.2.1 Inputs and Outputs

As is with the physical model formulation, input $x(n)$ at a point i, j of the mesh is achieved by adding $\frac{1}{2}x(n)$ to neighboring ports $N_{i+1,j}(n)$, $S_{i-1,j}(n)$, $E_{i,j+1}(n)$, $W_{i,j-1}(n)$. Similarly, the output $y(n)$ at point i, j is one-half the sum of outgoing ports $N_{i,j}(n)$, $S_{i,j}(n)$, $E_{i,j}(n)$, $W_{i,j}(n)$. Because of the condition from [3] that the sum of the incoming waves at time $n-1$ is equal to the sum of outgoing waves at

n , it is possible to apply input to the mesh in the same fashion as output. This simplification can allow for simpler implementation in practice.

$$\begin{aligned}
N_{i,j}(n) &= N_{i,j}(n) + \frac{1}{2}x(n) \\
S_{i,j}(n) &= S_{i,j}(n) + \frac{1}{2}x(n) \\
E_{i,j}(n) &= E_{i,j}(n) + \frac{1}{2}x(n) \\
W_{i,j}(n) &= W_{i,j}(n) + \frac{1}{2}x(n)
\end{aligned} \tag{2.7}$$

$$y_{i,j}(n) = N_{i,j}(n) + S_{i,j}(n) + E_{i,j}(n) + W_{i,j}(n) \tag{2.8}$$

2.3 Comparison to Other Methods

The waveguide mesh is not the only approach to discretely modeling wave propagation in two (or higher) dimensions. Discretizations of the wave equation, such as the Finite Difference Time Domain (FDTD) method, are used to model membranes as well as acoustic propagation in 2D and 3D space; as covered in [5], [6], [7], [8], and [9]. Below is one formulation of the FDTD, based on the evaluation of each wave variable $u_{i,j}$ of the grid. Since the method only maintains a grid of the u variable, input and output is simply a matter of adding to or reading from the wave at a desired point.

$$\begin{aligned}
u_{i,j}(n+1) = \\
Cu_{i,j}(n) + D(u_{i-1,j}(n) + u_{i+1,j}(n) + u_{i,j-1}(n) + u_{i,j+1}(n)) - u_{i,j}(n-1)
\end{aligned} \tag{2.9}$$

In the FDTD scheme, C and D are parameters dependent on the spatial and time sampling intervals (Δx , Δy , Δt) and speed of wave propagation (c). While FDTD schemes tend to use these values as design parameters (provided they satisfy the Courant-Freidichs-Lewy condition), [3] demonstrates that the digital waveguide mesh can be viewed as a specific case of discretization of the wave equation with $\Delta x = \Delta y = \Delta t$ and the speed of wave propagation $c = \frac{1}{\sqrt{2}}$.

A chapter in [10] and other papers such as [11] discuss the equivalence of the FDTD and Waveguide mesh methods via a state transformation, and how a model might interconnect components that implement each method. The modular flexibility of boundary conditions on the waveguide mesh turn out to be one of its strengths compared to the FDTD, where even simple reflections can not be implemented so intuitively. Thus, FDTD approaches such as [12] have employed the discrete waveguide to FDTD "K-W" converter described in [10] to allow more complex boundary behavior from the discrete waveguide mesh to be employed in a FDTD scheme.

The pros and cons of discrete waveguide modeling versus the FDTD method are covered in [11]. To summarize: the numerical robustness of waveguide modeling is advantageous in 1D as it better supports time varying parameters and fractional delays useful for musical tuning. In 2D and higher dimensions, the waveguide mesh remains more robust when nonlinear boundaries or time varying parameters are introduced, but the FDTD uses less memory and operations per junction than the waveguide mesh. The performance advantage of the FDTD is not without the cost of numerical accuracy and even some instability near DC and the Nyquist frequency. It can be argued that in spite of the difference in number of operations per junction, the algorithmic complexity of both schemes, intuitively, is still ultimately bounded by $O(n^2)$.

2.3.1 Mesh Analysis and Alternative Representations

Like the goal of this thesis, existing research has sought after approaches to analyze the waveguide mesh and other discretization schemes with the intent of deriving a more condensed representation in the interest of performance. Using high performance computing techniques, [13], along with work by Bilbao and Webb ([7], [8], [9], [14], [15]), formulates FDTD schemes in a manner that is conducive to GPU and parallel architecture implementation. [16] presents a strategy for decimating the FDTD representation in the interest of improved real-time computation. Lastly, time domain analytical expressions for the 2D rectangular mesh are derived [17].

The square or rectangular 2D mesh is not the only approach to the digital waveguide model. Plenty of existing research has covered different topologies of the waveguide mesh ([18], [19]), as well applications of the 3D waveguide model ([12], [20]), where (in the rectangular case) each scattering junction has six ports as opposed to four. A paper from Bilbao, [14] provides an excellent survey of the practical issues of implementing discretization of the wave equations: from grid representations, algorithm design, GPU implementation and applications. While the 2D square mesh will be the primary focus of this paper, the methods presented can theoretically extend to higher dimensionality, different topologies, other time stepping methods, and rectangular meshes.

2.4 Implementation and Performance

The most widely available implementation of the mesh is available as C++ code in the STK mesh2D class [21]. It has been equivalently implemented in Matlab by Scavone in [1]. This implementation follows the two pass parallel strategy outlined in [3]: two copies of the mesh representation are stored. Each pass of the grid reads one 2D array containing traveling waves at time $n - 1$ and fills the second array with the result at time n . On the next iteration, the second array is read from while the first is updated. The benefit of maintaining two arrays of state is that it allows each scattering junction port to be evaluated independently. If only one array is used, some intermediate copying mechanism is needed to avoid updating a port before it has been read by it's neighbor. While it stands to reason that all the scattering junction ports could be evaluated concurrently, parallel solutions for the waveguide model have not been explored to the same degree as the FDTD method. It is also worth noting that in the STK or any other implementation, the state of $N_{i,j}^+$, $S_{i,j}^+$, $E_{i,j}^+$, $W_{i,j}^+$ need not be maintained, though $V_{i,j}$ variables are still often stored.

A paper from 2001, [4], provides benchmarking on real-time mesh performance. A Pentium III 500Mhz processor was merely capable of evaluating a 6x6 waveguide mesh in real-time using all of it's resources, while an Athlon 1.3GHz

system was only capable of a 10x10 mesh. At the outset of the research related to this paper, the STK mesh2D implementation ([21]) was benchmarked on a more modern Core i5 3.40GHz system, and was found to only be capable of evaluating a 55x55 sized mesh in real-time, using most of the system resources. A Core i5 2.20 GHz system could support a 45x45 mesh. As mentioned above, the $O(n^2)$ growth of the mesh ultimately keeps larger mesh sizes where more interesting sounds exist out of reach for real-time. This limitation has provided the motivation to investigate more efficient representations of the two dimensional waveguide mesh.

Chapter 3

The Mesh as a Transfer Function

3.1 Mesh Symmetry

One of the initial observations that motivated the work in [2] was that placing input and output at the center of the mesh, for odd sizes, results in symmetric wave prorogation. The consequence of this observation is that the computation of the mesh for that specific input output location can be reduced to that of a triangle roughly an eighth the original square. Thus, an M by M mesh (for odd M) reduces to a $\frac{M+1}{2}$ by $\frac{M+1}{2}$ triangle. Matrix (3.1) illustrates this reduction in the gray shaded area.

$$\begin{array}{ccccccc}
 J_{1,1} & J_{1,2} & \cdots & J_{1,\frac{M+1}{2}} & \cdots & J_{1,M-1} & J_{1,M} \\
 J_{2,1} & J_{2,2} & \cdots & J_{2,\frac{M+1}{2}} & \cdots & J_{2,M-1} & J_{2,M} \\
 J_{3,1} & J_{3,2} & \cdots & J_{3,\frac{M+1}{2}} & \cdots & J_{3,M-1} & J_{3,M} \\
 \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
 J_{\frac{M+1}{2},1} & J_{\frac{M+1}{2},2} & \cdots & J_{\frac{M+1}{2},\frac{M+1}{2}} & \cdots & J_{\frac{M+1}{2},M-1} & J_{\frac{M+1}{2},M} \\
 \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
 \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\
 J_{M,1} & J_{M,2} & \cdots & J_{M,\frac{M+1}{2}} & \cdots & J_{M,M-1} & J_{M,M}
 \end{array} \tag{3.1}$$

3.1.1 Difference Equation

It is possible to express the triangle section of the mesh with center excitation as difference equations based on section 2.2 and leverage the performance improvement (at least for the specific case of center input and output) of only having to evaluate roughly an eighth of the mesh. As detailed in [2], the diagonal and center column elements now have their own single sample boundary conditions. Corner elements will clearly possess two boundary conditions. The center column eastern boundary input is equivalent to the center column western junction input:

$$E_{i, \frac{M+1}{2}}(n) = W_{i, \frac{M+1}{2}+1}(n-1) = E_{i, \frac{M+1}{2}-1}(n-1) \quad (3.2)$$

Along the diagonal, the southern boundaries are equal to the eastern port input, and the western boundary is equal to the northern port input.

$$\begin{aligned} S_{i,i}(n) &= N_{i+1,i}(n-1) = W_{i,i+1}(n-1) \\ W_{i,i}(n) &= E_{i,i-1}(n-1) = S_{i-1,i}(n-1) \end{aligned} \quad (3.3)$$

3.1.2 Performance Timings

Table 3.1 provides performance timings of how much of an improvement the triangular difference equations offer over the STK square mesh implementation. The triangular difference scheme is most easily implemented using a two dimensional array of size $\frac{M+1}{2}$ by $\frac{M+1}{2}$. The extra memory overhead is a trade off of easier array indexing. Though the mesh with center excitation performance still grows in $O(n^2)$, running on an roughly an eighth the number of junctions as the full size mesh does allow for much larger mesh sizes to be computed in real-time.

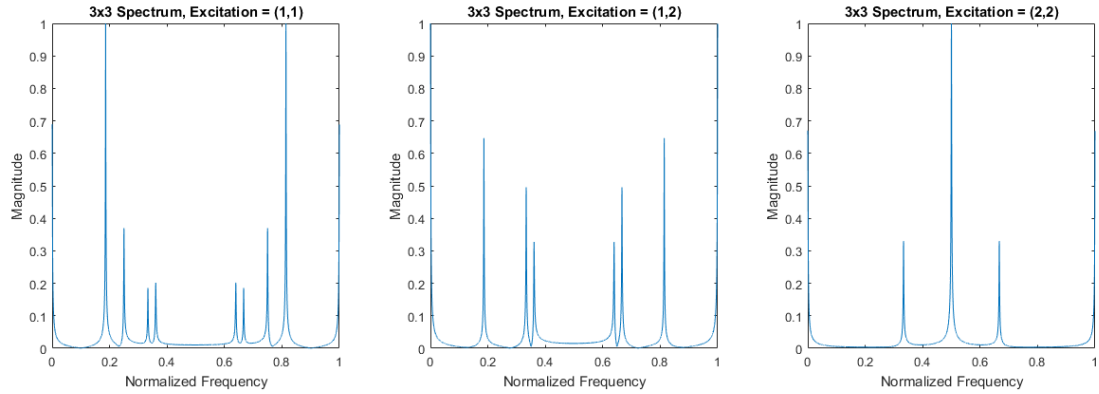
3.1.3 Constrained Input/Output

While the symmetric reduction does allow larger mesh sizes to be computed, it gives up the free parameter of the physical model to select input and output locations. It is empirically observed that the center excitation that allows for this reduction produces the simplest possible timbre, as is evidenced by comparing the

Table 3.1: Benchmarking of difference equations for mesh with center excitation

Size	STK Mesh (ms)	Center Excitation(ms)
9x9	27.9	3.6
19x19	116.2	15.7
29x29	266.1	38.2
39x39	507.2	71.2
49x49	783.6	110.2
59x59	1140.6	160.2

spectrum of the three possible excitation points of the 3x3 mesh (3.1). This observation coincides with the intuition of a waveguide string model in one dimension, where an input in the middle excites the fewest possible vibrational modes, or causes cancellation in the most amount of modes.

**Figure 3.1:** Normalized spectra of 3x3 mesh for different excitations

3.2 Transfer Function Analysis

The motivation to derive a transfer function $H(z)$ from the waveguide mesh is twofold. First, the transfer function can be implemented as an infinite impulse response (IIR) filter, with the expectation that it can synthesize equivalent sound as the waveguide model using a smaller representation and less computation. Second,

possessing an analytical representation of the mesh can allow for more insightful observation of the mesh's behavior (such as inspection of the zeros and poles of the system).

3.2.1 Symbolic Computation

Transfer function analysis can be applied to the mesh for a static border reflection $\mathbf{a}$, such as $\mathbf{a} = 1$ or $\mathbf{a} = 0.99$. In this case, the value of $\mathbf{a}$ is embedded in the numeric coefficients of $H(z)$:

$$H(z) = \frac{B(z)}{A(z)} = \frac{b_0 + b_1 z^{-1} + \cdots + b_{n-1} z^{-n-1} + b_n z^{-n}}{a_0 + a_1 z^{-1} + \cdots + a_{m-1} z^{-m-1} + a_m z^{-m}} \quad (3.4)$$

What might be more useful is to perform the analysis while maintaining $\mathbf{a}$ as a variable, allowing $H(z)$ to take the form of an analytical solution, where each coefficient is a function of the boundary reflection $\mathbf{a}$. This would allow the boundary reflections of the mesh to be controllable parameters within the transfer function. Reflection filters can also be implemented in $H(z)$ by substituting $\mathbf{a}$ with the frequency domain representation of a filter $\mathbf{a}(z)$.

$$H(z) = \frac{f_0^b(\mathbf{a}) + f_1^b(\mathbf{a})z^{-1} + \cdots + f_{n-1}^b(\mathbf{a})z^{-n-1} + f_n^b(\mathbf{a})z^{-n}}{f_0^a(\mathbf{a}) + f_1^a(\mathbf{a})z^{-1} + \cdots + f_{m-1}^a(\mathbf{a})z^{-m-1} + f_m^a(\mathbf{a})z^{-m}} \quad (3.5)$$

A symbolic computing environment, such as Matlab's Symbolic Math Toolbox, or Wolfram Mathematica, is necessary to implement transfer function analysis for either case, since the variable z needs to be maintained in either case, and the variable $\mathbf{a}$ for the case of (3.5). Maintaining $\mathbf{a}$ however, adds a level of complexity to the representation of $H(z)$.

3.2.2 Center Excitation

[2] uses the symmetrically reduced difference equations from section 3.1.1 as a first approach at deriving the transfer function from the simplified formulation of the mesh. Smyth's algorithm takes the Z-transform of the difference equation formulation, and procedurally substitutes and eliminates all junction port variables $(N_{i,j}, S_{i,j}, E_{i,j}, W_{i,j})$ until an expression for $Y(z)$ is reached, solely dependent

on the input $X(z)$. At each step of the algorithm, the rational function must be expanded and common factors of the numerator and denominator canceled. Finally, by dividing output over input, $\frac{Y(z)}{X(z)}$, the waveguide mesh is successfully converted to a transfer function $H(z)$.

An implementation of the algorithm using Matlab's Symbolic Toolbox was used to derive the transfer function of the 5x5 triangle corresponding to the 9x9 mesh excited at the center. The result, in line with (3.5), produces an transfer function where each a_i and b_i coefficient is a polynomial in terms $\mathbf{a}$. In spite of the result, the slow performance of Matlab's Symbolic Toolbox (taking over 17 hours to compute the 5x5 transfer function for center excitation on the Core i5 platform), as well as the limitations of timbral possibilities of the center excitation, motivates investigation into transfer function analysis for corner excitation (where the most amount of mode excitation occurs) using a different environment.

3.2.3 Square Mesh with Corner Excitation

The transfer function analysis method can be extended to the square mesh with corner excitation by taking the z-transform of the difference equations in 2.2. Like the center excitation case, the algorithm traverses the grid, continually substituting, eliminating and reducing variables until a transfer function is reached. Each row of the mesh is iterated over, with the top and bottom rows being special cases due to boundary conditions. Within each row, an inner loop iterates over each non-border element, while the first and last element are treated as special boundary cases. Horizontal symmetry allows some substitution steps to be simplified to a copy or re-ordering of some previously reduced variable, significantly reducing computation time.

The transfer function analysis algorithm for corner excitation was implemented in Wolfram Mathematica as part of the research for this thesis. Table 3.2 collects the results of the Mathematica implementation on the i5 platform: analysis time, resulting filter order, and the C++ timing of a direct form II IIR filter of equivalent size. The C++ timing of the corresponding STK mesh size is also provided for reference. The C++ DFII timing is theoretical based on the size

Table 3.2: Results of Mathematica transfer function analysis

Size	Analysis Time	IIR Order (Num/Den)	DFII (ms)	Mesh (ms)
3x3	630 ms	19 / 20	1.2	3.9
4x4	6 sec	31 / 32	1.7	6.4
5x5	36 sec	45 / 46	2.5	9.5
6x6	3 min	61 / 62	3.3	13.5
7x7	15 min	79 / 80	4.4	17.8
8x8	70 min	99 / 100	5.2	22.8
9x9	3 hrs	121 / 122	6.3	27.9
10x10	11.5 hrs	145 / 146	7.2	34.9
11x11	26 hrs	171 / 172	8.2	41.8

of the IIR filter; the audio result is not validated for correctness. The following section will outline why the transfer function is not yet usable for audio synthesis.

While noticeably faster at symbolic computation than Matlab, the exponential growth of analysis time still dominates the performance Mathematica provides. Only small mesh sizes were able to be analyzed in a tractable amount of time. The filter order’s stable growth does appear promising; it is possible to infer that were the transfer function for a much larger mesh computable, the result would be a IIR filter order of reasonable size.

3.3 Usage of Symbolic Transfer Function

The explosive cost of transfer function analysis is not the only issue that severely restricts the usage of the symbolic transfer function in it’s canonical form ((3.4) and (3.5)). After directly implementing the symbolic IIR filter as code for reflection coefficient $\mathbf{a} = 1$, inspection of the zeros and poles reveal profound instability for increasing mesh sizes, as seen in figure 3.2. Even the 6x6 and 7x7 exhibit small amounts of instability. Though the transfer function for $\mathbf{a} = 1$ is expected to be marginally stable, smaller values of $\mathbf{a}$ (still in the range of useful values for the boundary reflections, such as $\mathbf{a} = 0.9$) exhibit instability as well.

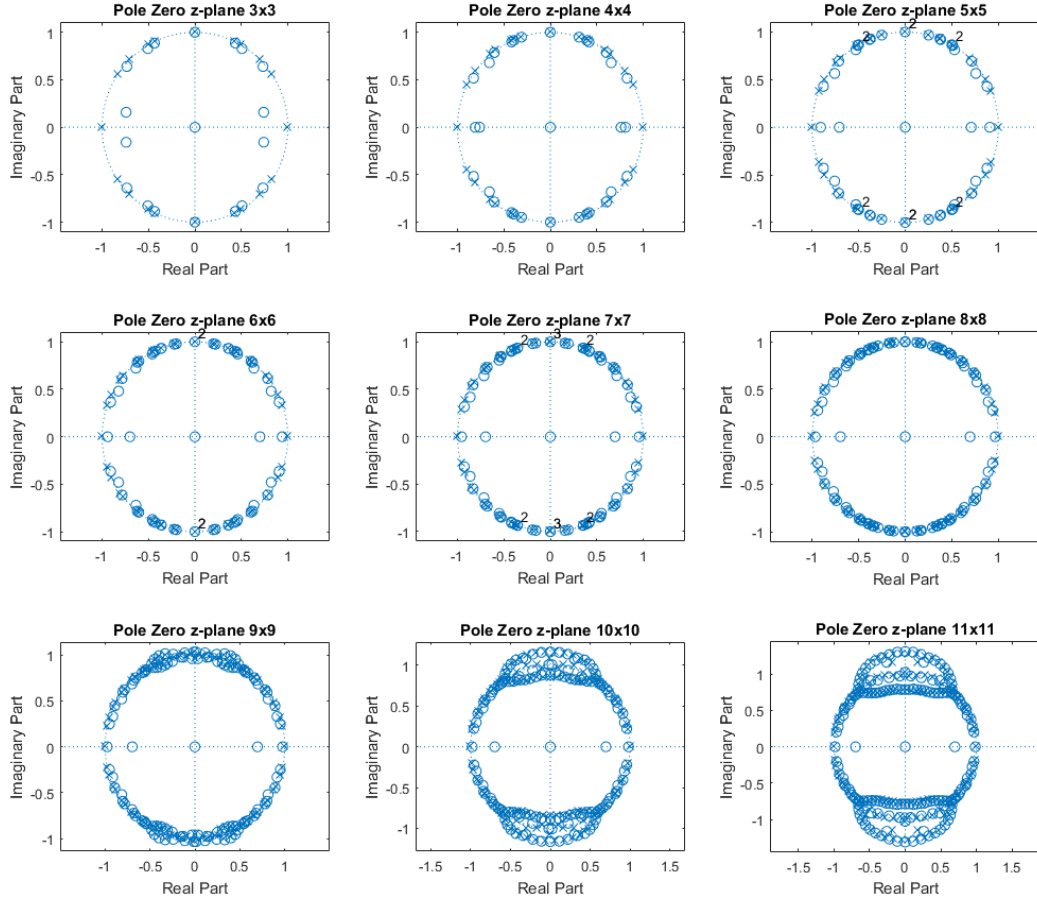


Figure 3.2: Zero pole plots from truncated symbolic transfer functions, $\mathbf{a} = 1$

3.3.1 Coefficient Quantization

One of the other properties of working in a symbolic computing environment is that numbers can (and by default *are* in Mathematica) represented with arbitrary precision arithmetic, allowing for digit precision only limited by system memory. Some of the coefficients of a resulting transfer function, such as $f_{170}^b(\mathbf{a})$ of the 11x11, contain terms in $\mathbf{a}$ with greater than 16 digits of precision, the limit of digit precision is a 64 bit floating point environment ([22]).

$$f_{170}^b(\mathbf{a}) = -20266198323167232\mathbf{a}^{19} - 38843546786070528\mathbf{a}^{20} + 25614222880669696\mathbf{a}^{21}$$

In order to make use of the transfer function for audio processing, coefficients will inevitably be truncated to fixed precision. Intuitively, running the analysis for a fixed value of $\mathbf{a}$, or evaluating $f_{170}^b(a)$ to some value, does not alleviate this issue. The coefficients of $H(z)$, or the terms of $\mathbf{a}$ of each coefficient, are being quantized. Though hypothetical 128-bit computers may allow for up to 36 digits of precision to represent the 11x11 transfer function accurately, it stands to reason that some larger mesh might even exceed expanded fixed precision limitations.

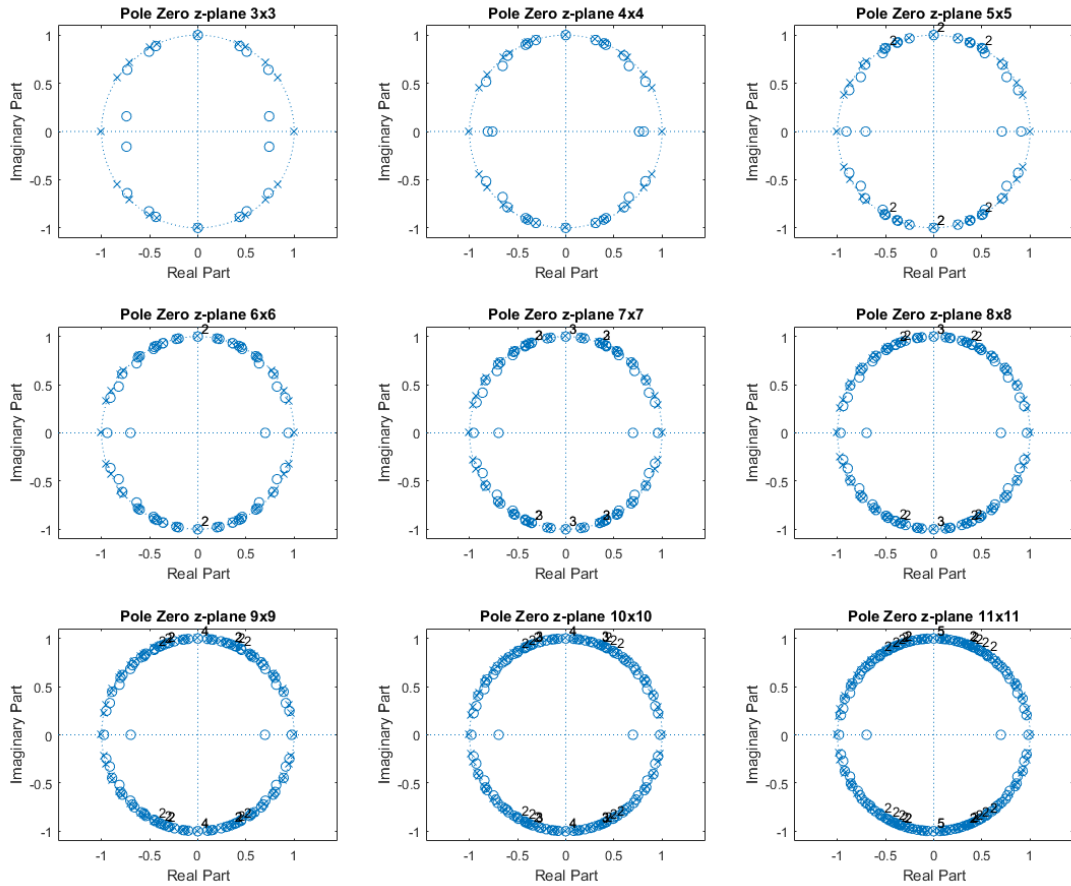


Figure 3.3: Zero pole plots with truncated zeros and poles, $\mathbf{a} = 1$

The effects of coefficient quantization on various digital systems is well detailed by Oppenheim in [23]. The problem pertinent to the symbolic transfer function here is that the canonical form which is being solved for is also the most vulnerable to coefficient quantization. With any large order expanded polynomial (the

numerator $B(z)$ and denominator $A(z)$), even the slightest perturbations to any one coefficient can drastically displace the polynomial's roots: the zeros and poles of $H(z)$. The instability observed in 3.2 is a direct result of this phenomenon.

The good news is that other representations of $H(z)$ are more robust to quantization. In an arbitrary precision environment, the transfer function can be evaluated for a reflection coefficient value, and the the zeros and poles of the resulting system can be recovered. The zeros and poles can then be truncated to fixed precision, and produce a stable system. Figure 3.3 shows the zero pole plots of each transfer function, with truncation on the zeros and poles as opposed to the transfer function coefficients.

3.3.2 Second Order Sections

Zeros and poles can not be used to directly implement a digital filter. They can, however, be used to compose a bank of real second order IIR filter sections. Like the stability of the factored zero pole representation, small perturbations from truncation in one second order section does not affect other sections. Second order sections (SOS) can either be implemented in parallel (as a sum) ¹:

$$H(z) = g \sum_{i=1}^N H_i(z) = g \sum_{i=1}^N \frac{b_{i,0} + b_{i,1}z^{-1}}{a_{i,0} + a_{i,1}z^{-1} + a_{i,2}z^{-2}} \quad (3.6)$$

Or in series (as a product):

$$H(z) = g \prod_{i=1}^N H_i(z) = g \prod_{i=1}^N \frac{b_{i,0} + b_{i,1}z^{-1} + b_{i,2}z^{-2}}{a_{i,0} + a_{i,1}z^{-1} + a_{i,2}z^{-2}} \quad (3.7)$$

Partial fraction expansion ([24]) can be used to convert the canonical transfer function to a parallel representation, while series form can be derived by matching complex conjugate zero pairs and pole pairs (notice that in figure 3.3, each zero pole plot is symmetric), and then pairing pole zero pairs in proximity to each other. The function `residuez` can be used in Matlab to perform partial fraction expansion, and real second order sections can be assembled out of the complex conjugate one pole sections. Unfortunately, implementing partial fraction expansion according

¹where g is gain

Table 3.3: Results of C++ implementation of series second order sections

Size	IIR Order	Num of SOS	SOS (ms)	Mesh (ms)
3x3	19 / 20	10	2.0	3.9
4x4	31 / 32	16	3.2	6.4
5x5	45 / 46	23	4.7	9.5
6x6	61 / 62	31	6.3	13.5
7x7	79 / 80	40	7.7	17.8
8x8	99 / 100	50	9.8	22.8
9x9	121 / 122	61	11.4	27.9
10x10	145 / 146	73	13.3	34.9
11x11	171 / 172	86	15.8	41.8

to [24] (as `residuez` does) produces filters that suffer from the same coefficient quantization issues discussed in section 3.3.1. Matlab’s `zp2sos` function provides an implementation that converts zero pole representations into a series of second order sections that does not suffer from such issues.

Table 3.3 contains the results of implementing series second order sections in C++. The SOS performance times are larger than hypothetical transfer function times in table 3.2, but that is understandable given the observation that an N th order IIR filter can be converted to $\frac{N}{2}$ second order sections, and each second order section contains 3 feed forward (B) and feed backward elements (A). The SOS implementation should require approximately 1.5 times the operations. In spite of the incurred cost of the second order section representation, it still performs much faster than the conventional physical model implementation.

Unfortunately, neither symbolic zeros and poles nor second order sections can be represented in any reducible and useful form when derived from the canonical transfer function output of the symbolic analysis algorithm. For all but the simplest case (3x3 excited in the center), Mathematica is unable to express all poles and zeros as radicals, and instead expresses them as *Root* functions of the original transfer function polynomial; clearly not usable in a real-time filter implementation. This limitation renders the original goal of maintaining the physical model’s

free parameter of boundary reflections not yet achievable. Taking into account the explosive computational expense of symbolic analysis as well, other methods for converting the two dimensional mesh into a filter (or bank of filters) are explored in chapter 4.

Chapter 4

State Space Representation of Mesh

4.1 State Space Model

[10] demonstrates that the digital waveguide (and equivalently the FDTD scheme) can be represented as a matrix transformation and thus is eligible for state space analysis. Indeed, research from Bilbao ([14], [15]) has commonly framed discrete time stepping methods as matrix operations that are conducive to analysis. For the 2D waveguide, the state of all $N_{i,j}$, $S_{i,j}$, $E_{i,j}$, $W_{i,j}$ grids can be combined into a linearized vector $\underline{\mathbf{u}}$ of total length $4M^2$, where M is the dimension of the mesh. Input and output vectors B and C contain nonzero elements $\frac{1}{2}$ at the four $N_{i-1,j}$, $S_{i+1,j}$, $E_{i,j+1}$, $W_{i,j-1}$ positions corresponding to the outgoing junctions for a given $J_{i,j}$ (as per section 2.2.1) in $\underline{\mathbf{u}}$. Given the realization in section 2.2.1 ([3]) that the incoming junctions are equivalent to the outgoing junctions after one time step, it is equivalent to select the incoming junctions instead ($N_{i,j}$, $S_{i,j}$, $E_{i,j}$, $W_{i,j}$), allowing the four nonzero elements to be adjacent to one another in B and C .

$$\underline{\mathbf{u}} = \begin{bmatrix} \vdots \\ N_{i,j} \\ S_{i,j} \\ E_{i,j} \\ W_{i,j} \\ \vdots \end{bmatrix} \quad B = \frac{1}{2} \begin{bmatrix} \vdots \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix} \quad C = \frac{1}{2} \begin{bmatrix} \vdots \\ 1 \\ 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix} \quad (4.1)$$

The state transition matrix A is a $4M^2$ by $4M^2$ sparse matrix where each row contains four non zero elements equal to $\frac{1}{2}$ whose positions correspond to the linear combinations of elements from $\underline{u}$ according to equation (2.6). Note the grouping of four rows with similar non zero columns that correspond to each $N_{i,j}$, $S_{i,j}$, $E_{i,j}$, $W_{i,j}$.

[illegible]

The equations in appendix A characterize the reflection behavior at the boundary with a 2 sample round drip delay and coefficient $\mathbf{a}$. To avoid having to maintain the state of $\underline{\mathbf{u}}$ at two time steps, the state vector $\underline{\mathbf{u}}$ and matrix A can be extended to include state $B_n^{N,S,E,W}$ for the round trip delay at each border. C and D simply need to be zero padded to the appropriate length.

[illegible]

A border row in A now contains a single $\mathbf{a}$ at a position corresponding to a boundary junction variable $(N, S, E, W)_{i,j}$ in $\underline{\mathbf{u}}$. The original rows in A corresponding to a boundary junction position will now contain some term in it's linear combination at a position corresponding to the $B_i^{N,S,E,W}$ index in $\underline{\mathbf{u}}$. For an M by M waveguide mesh, $\underline{\mathbf{u}}$, B and C are now vectors of length $4M^2 + 4M$, and A is of size $4M^2 + 4M$ by $4M^2 + 4M$. The state of the mesh can then be expressed as a function of input $x(n)$:

$$\underline{\mathbf{u}}(n+1) = A\underline{\mathbf{u}}(n) + Bx(n) \quad (4.4)$$

And the output $y(n)$ can be expressed in terms of the transpose of C :

$$y(n) = C^T \underline{\mathbf{u}}(n) \quad (4.5)$$

It goes without saying that the state space model presented here is probably the least efficient way to implement the waveguide mesh, given the size and sparsity of A . However, the state space formulation will allow for an intuitive derivation of the transfer function or system zeros and poles.

4.2 State Space Analysis

Following [24], the impulse response of the mesh $H(z)$ can now be expressed in terms of A , B and C :

$$H(z) = C(zI - A)^{-1}B \quad (4.6)$$

$H(z)$ can be separated into it's numerator $B(z)$ and it's denominator $A(z)$ given that the the inverted matrix $(zI - A)^{-1}$ can be expressed as the quotient of the adjoint and the determinant of $(zI - A)$. The denominator $A(z)$ a of $H(z)$ can be expressed as:

$$A(z) = \det(zI - A) \quad (4.7)$$

While the numerator $B(z)$ can be expressed in terms of the adjoint matrix.

$$B(z) = C \text{adj}(zI - A)B \quad (4.8)$$

Equation (4.7) is also the expression for the characteristic polynomial of A , thus the roots z_i of $A(z)$, or the poles of $H(z)$, are the eigenvalues of the matrix A . The characteristic polynomial and eigenvalues of A can be extracted analytically from equation (4.7). Similarly, the roots of equation (4.8) are the zeros of $H(z)$.

Unlike the transfer function analysis in chapter 3, zero pole cancellation is not embedded in state space analysis. Thus, the result of analysis is a transfer function of order $4M^2 + 4M$, or a set of zeros and poles each of size $4M^2 + 4M$. Zero pole cancellation can be applied by simply removing equivalent zeros and poles from the numerator $B(z)$ and denominator $A(z)$.

4.2.1 Insights about Mesh Transfer Function

State space analysis also reveals some theoretical insights for empirical observations made about the transfer function derived in chapter 3.

- The maximum possible order of a transfer function for a given mesh size without zero pole cancellation is $4M^2 + 4M$, the corresponding size of $\underline{u}$.
- The poles of the mesh are dependent exclusively on the state transition matrix A , thus the poles of different excitations on a mesh of size M are subsets for of the set of eigenvalues of A .
 - It was observed that the poles of the center excitation point were a subset of the edge excitation poles, which were a subset of the center excitation poles.
- The zeros of the system are dependent on the input and output points (vectors B and C). Accordingly, the amount of zero pole cancellation of a transfer function is also based on the chosen excitation point.

4.3 Analysis in a Fixed Precision Environment

The eigenanalysis on A can be performed on symbolic or real values of $\mathbf{a}$. For the symbolic case, implementation in Mathematica of transfer function analysis

from chapter 3 vastly outperforms state space analysis. However, since the state space model need not carry a representation of z , numeric analysis implemented in a fixed precision environment such as Matlab performs much faster, albeit for a fixed $\mathbf{a}$.

Since state space analysis is best implemented in a fixed precision environment, the coefficient quantization problem discussed in section 3.3.1 needs to be avoided. The benefit of matrix decomposition techniques such as the QR algorithm is twofold: they solve for the eigenvalues directly, avoiding the need to factor the characteristic polynomial where quantization can occur on it's coefficients, and they generally perform faster for numeric solutions than the analytical equations (4.7) and (4.8).

Control systems theory ([25], [26]) details other approaches on how to extract the zeros from a state space model. One such approach is through finding the roots of the determinant of the Rosenbrock system matrix ([27]).

$$\det \begin{bmatrix} zA - I & B \\ C & 0 \end{bmatrix} = 0 \quad (4.9)$$

This method of recovering the zeros of a system can be expressed equivalently as the finite solutions of the generalized eigenvalue problem for the matrices:

$$\begin{bmatrix} A & B \\ C & 0 \end{bmatrix}, \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix} \quad (4.10)$$

Like numeric eigenvalue algorithms, the QZ algorithm ([28]) can be used to solve the generalized eigenvalue problem directly and quicker than the analytical solution (4.8) involving the adjoint matrix.

Matlab provides functions to convert from state space to zero pole representations (and vice-versa) in `ss2zp` and `zp2ss`. Converting a state space model directly to a transfer function using `ss2tf` will create an unstable system due to coefficient quantization.

4.3.1 Implementation Results

State space models were analyzed in Matlab on the i5 platform and the generated number of zeros, poles and corresponding number of second order sections

Table 4.1: State space analysis and C++ timings - corner excitation

Size	Analysis Time	Order (Z/P)	Num SOS	SOS (ms)	Mesh (ms)
9x9	120 ms	121/122	61	11.3	27.9
19x19	4 sec	449/450	225	40.5	116.2
29x29	50 sec	955/956	478	85.7	266.1
39x39	5 min	1819/1820	910	190.1	507.2
49x49	16 min	2791/2792	1396	283.4	783.6
59x59	43 min	4115/4116	2058	413.7	1140.6

(SOS) were gathered in table 4.1. As was performed in chapter 3, the second order sections for larger meshes were implemented in C++ and their performances timed against the STK implementation. Table 4.1 shows that the analysis time is much faster for the numerical scenario, allowing for the zeros, poles and second order section representations of much larger meshes to be derived. The trade off, as discussed before, is that analysis can only be performed on fixed boundary coefficient values. Like table 3.3, the C++ implementation of second order sections continues to perform much faster than the conventional mesh implementation.

Since the state space analysis can be performed for any input and output combination, table 4.2 collects the zero and pole counts and associated number of second order sections for excitation (equivalent input and output positions) at the middle edge and center positions. Given the observations from figure 3.1, it's no surprise that the center excitation produces the least amount of zeros and poles, and the corner produces the most. Table 4.3 presents the zero and pole orders for every possible input output combination on the 3x3 mesh. Note the symmetry about the triangle shaded in gray, indicating the set of unique possible excitation choices from all input output combinations. One observation to be made here is that all but two possible I/O combinations cancel more than half of the original 48 zeros and poles of the state space analysis. Even the largest case, 25/28, exhibits significant cancellation as well.

Table 4.2: Center and edge excitation: zero pole order and SOS count

Size	Center Z/P	Center SOS	Edge Z/P	Edge SOS
9x9	37/38	19	93/94	47
19x19	127/128	64	343/344	172
29x29	271/272	136	749/750	375
39x39	599/600	300	1395/1396	698
49x49	1463/1464	732	2339/2340	1170
59x59	1411/1412	706	3219/3220	1610

Table 4.3: Orders for every possible input output pair on the 3x3 mesh

I/O	(1,1)	(1,2)	(1,3)	(2,1)	(2,2)	(2,3)	(3,1)	(3,2)	(3,3)
(1,1)	19/20	14/16	25/28	14/16	5/8	12/16	25/28	12/16	15/20
(1,2)	14/16	15/16	14/16	21/24	6/8	21/24	12/16	13/16	12/16
(1,3)	25/28	14/16	19/20	12/16	5/8	14/16	15/20	12/16	25/28
(2,1)	14/16	21/24	12/16	15/16	6/8	13/16	14/16	21/24	12/16
(2,2)	5/8	6/8	5/8	6/8	7/8	6/8	5/8	6/8	5/8
(2,3)	12/16	21/24	14/16	13/16	6/8	15/16	12/16	21/24	14/16
(3,1)	25/28	12/16	15/20	14/16	5/8	12/16	19/20	14/16	25/28
(3,2)	12/16	13/16	12/16	21/24	6/8	21/24	14/16	15/16	14/16
(3,3)	15/20	12/16	21/24	12/16	5/8	14/16	21/24	14/16	19/20

4.4 Parallel Second Order Sections

The state space can also provide a more robust means of deriving the parallel second order sections representation without quantization error. After cancellation, the set of poles and zeros can be used to reconstruct a minimal state space model (`zp2ss` in Matlab). Then, [24] outlines how the transfer function $H(z)$ can be expressed as a sum of one pole sections based on the diagonalized state space representation. The minimal state space model $A_{min}, B_{min}, C_{min}$ can be diagonalized using the eigenvector matrix of A_{min} : E .

$$\begin{aligned}\tilde{A} &= E^{-1}A_{min}E \\ \tilde{B} &= E^{-1}B_{min} \\ \tilde{C} &= C_{min}E\end{aligned}\tag{4.11}$$

Of course, $\tilde{A}$ is simply a matrix with nonzero diagonal elements equal to the eigenvalues of A_{min} , which have already been stored as the poles p_i of the mesh¹. The new transfer function $\tilde{H}(z)$ can be expressed a sum of one pole sections, where c_i and b_i are the i th elements of $\tilde{C}$ and $\tilde{B}$, whose product is residue r_i :

$$\begin{aligned}\tilde{H}(z) &= \tilde{C}(zI - \tilde{A})^{-1}\tilde{B} \\ &= \sum_{i=1}^N \frac{\tilde{c}_i\tilde{b}_i}{1 - p_iz^{-1}} = \sum_{i=1}^N \frac{r_i}{1 - p_iz^{-1}}\end{aligned}\tag{4.12}$$

The sections corresponding to complex conjugate pole pairs p_i and $\bar{p}_i$ can be combined to form second order real sections. In practice, at least for the square mesh, there are always two one pole sections with real residues and poles r_1, r_2, p_1 and p_2 which form one final K th section.

$$\tilde{H}(z) = \sum_{i=1}^{K-1} \frac{2\operatorname{Re}(r_i) - 2\operatorname{Re}(r_i\bar{p}_i)z^{-1}}{1 - 2\operatorname{Re}(p_i)z^{-1} + |p_i|^2z^{-2}} + \frac{(r_1 + r_2) - (r_1p_2 + r_2p_1)z^{-1}}{1 - (p_1 + p_2)z^{-1} + p_1p_2z^{-2}}\tag{4.13}$$

Excluding the possibility of evaluating the filters concurrently, there is no expected performance benefit from the parallel section implementation over the series sections in table 4.1, as a given mesh size reduces to the same number of series or parallel sections, $K = \frac{N}{2}$.

¹Though to use them in (4.12) they need to be stored in the same order as the diagonal elements of $\tilde{A}$

Chapter 5

Approximation of Mesh Parameters

5.1 Effects of Varying the Reflection Coefficient

Inspection of resulting zeros and poles (of the 3x3 mesh in figure 5.1) for changing scalar reflection values reveals how $\mathbf{a}$ predominately affects the radius of the poles and zeros. This behavior can be used to reintroduce a boundary-like parameter to the numerical second order section (or zero pole) representation of the mesh, as described in chapter 4. Exploiting the polar coordinate form (radius r and angle θ) of complex zeros and poles, equation (5.1) demonstrates how a function of $\mathbf{a}$ can be multiplied against each zero and pole to change it's radius.

$$\begin{aligned} f(\mathbf{a})c &= f(\mathbf{a})(a + ib) \\ &= f(\mathbf{a})(r \cos \theta + i(r \sin \theta)) \\ &= (f(\mathbf{a})r)(\cos \theta + i \sin \theta) \end{aligned} \tag{5.1}$$

The best way to use equation (5.1) in practice is to apply it to the zeros and poles of the numerical state space analysis performed on the mesh for $\mathbf{a} = 1$, which of course produces the radii of 1 for all poles, and the maximum possible radii for zeros.

5.1.1 Choosing an Approximation Function

It is at least intuitively clear from figure 5.1 that the mapping from $\mathbf{a}$ to radius is not linear (as in $f(\mathbf{a}) = \mathbf{a}$). Figure 5.2 shows how decreasing values of $\mathbf{a}$

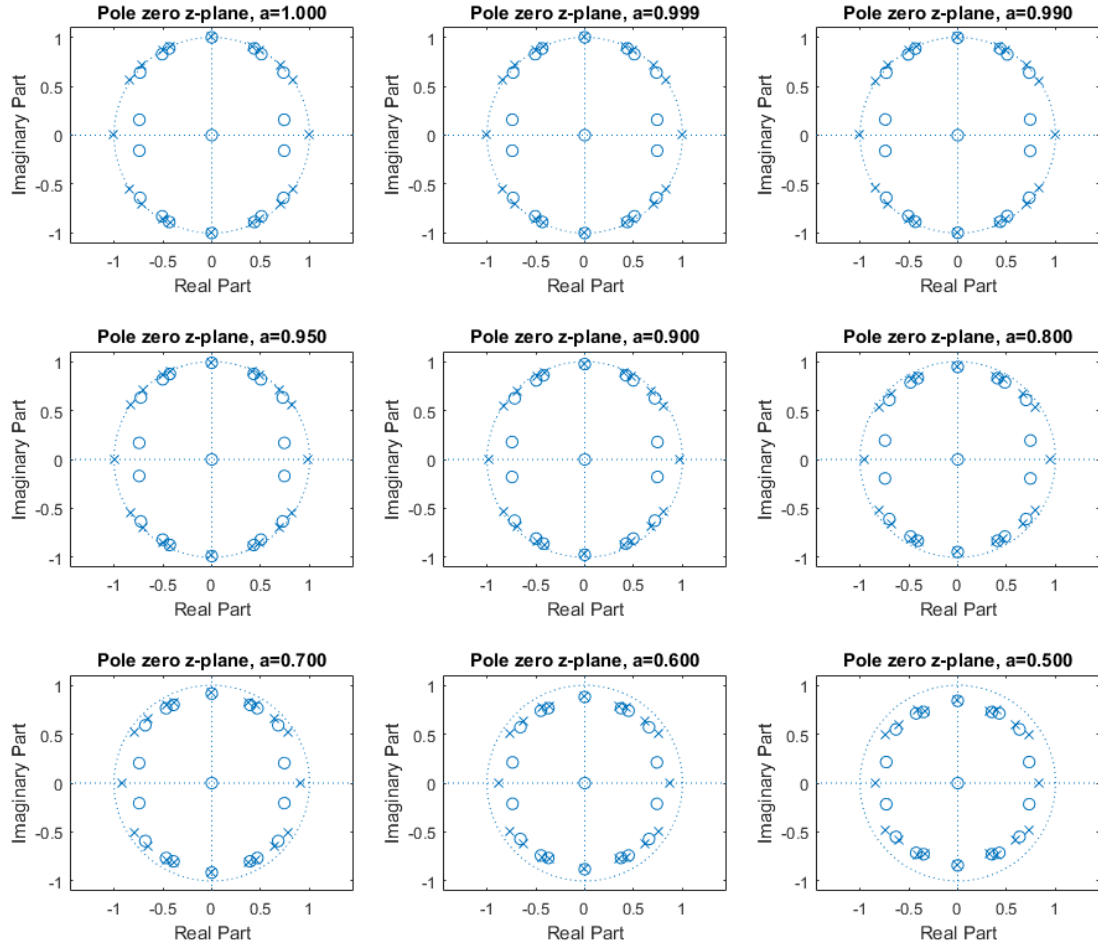


Figure 5.1: Pole zero plots of 3x3 for range of $\mathbf{a} = [0.5, 1]$

drastically sharpens the exponential decay envelope of the time domain waveform. Based on both these observations, an initial function that was empirically found to work well was $f(\mathbf{a}) = {}^{M+1}\sqrt{\mathbf{a}}$, based on the fact that it is an increasing function with a decreasing derivative in the range of $\mathbf{a} = [0.5, 1]$ that still satisfies $f(1) = 1$, scales with the mesh dimension M , and (though not pictured) yields amplitude envelopes for values of $\mathbf{a}$ very close to those seen in figure 5.2. Comparing the frequency responses of the actual and approximated boundary coefficients in figure 5.3 provides an idea how well this approximation performs. Understandably, greater differences occur as the approximated parameter moves away from $\mathbf{a} = 1$.

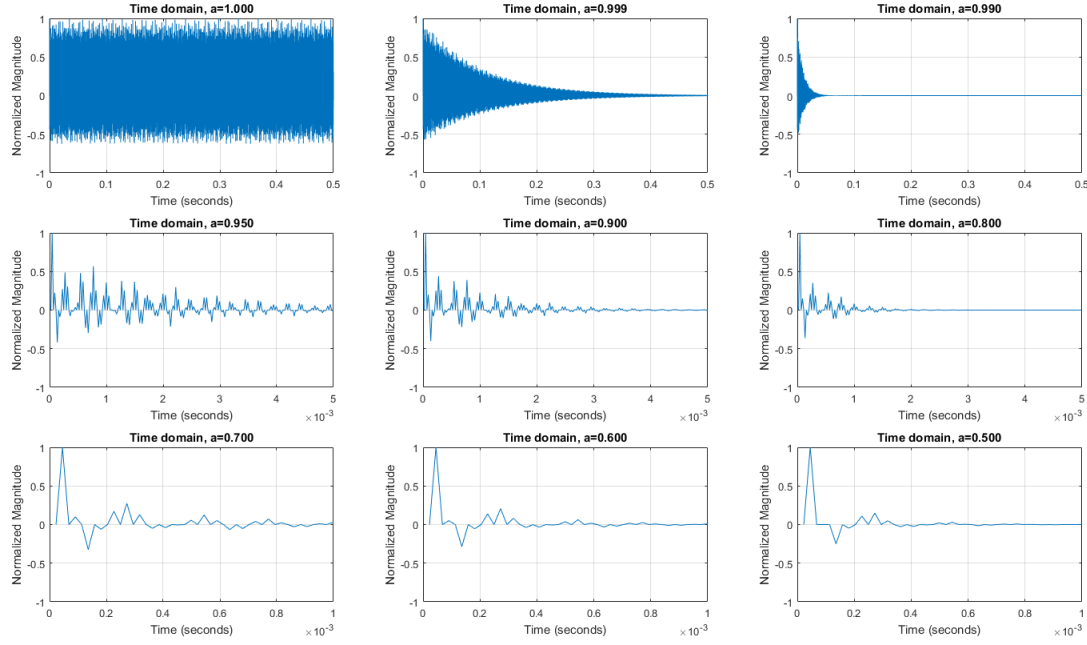


Figure 5.2: Time domain plots of 3x3 for range of $\mathbf{a} = [0.5, 1]$, note the decreasing time scale of the x-axis (time)

5.1.2 Second Order Sections

Series second order sections as covered in chapter 3 are useful for implementing approximated boundary loss. The coefficients of a section can be expressed in terms of the zeros, q_i and poles, p_i , that are paired with their complex conjugates when each biquadratic section is formed.¹

$$\begin{aligned}
 H(z) &= g \prod_{i=1}^N H_i(z) \\
 &= g \prod_{i=1}^N \frac{b_0 + b_{i,1}z^{-1} + b_{i,2}z^{-2}}{1 + a_{i,1}z^{-1} + a_{i,2}z^{-2}} \\
 &= g \prod_{i=1}^N \frac{b_0 - 2 \operatorname{Re}(q_i)z^{-1} + |q_i|^2 z^{-2}}{1 - 2 \operatorname{Re}(p_i)z^{-1} + |p_i|^2 z^{-2}}
 \end{aligned} \tag{5.2}$$

Following equation (5.1), the $f(\mathbf{a})$ function applied to the zeros and poles can also be applied to the coefficients of each section. This is useful because the $f(\mathbf{a})$ can

¹For the series second order section containing a singular real zero, b_0 is equal to 0, otherwise it is equal to 1.

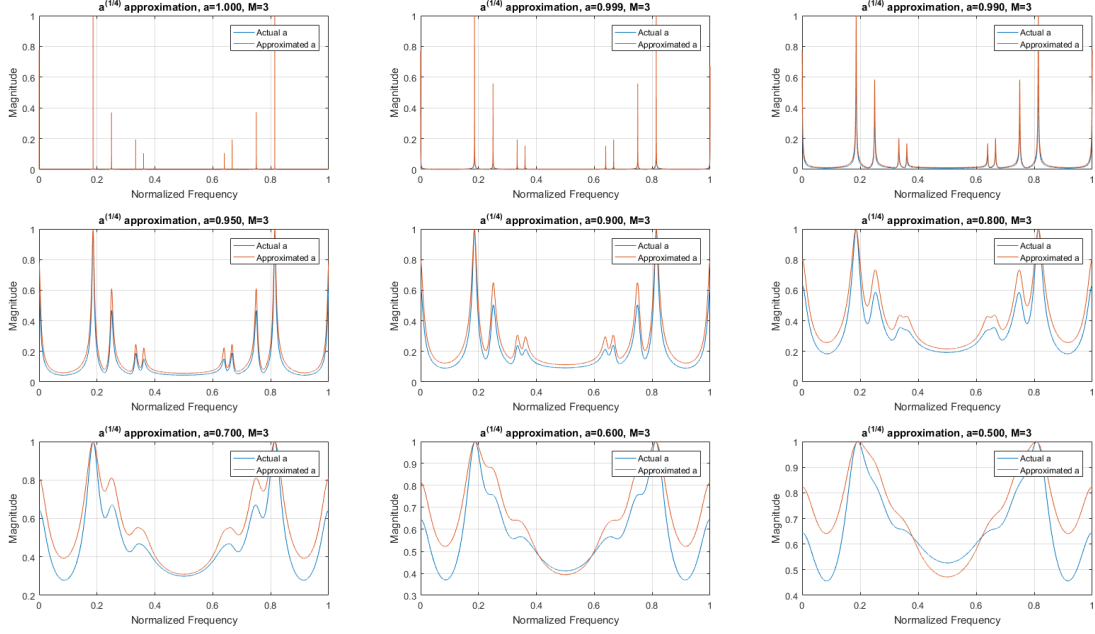


Figure 5.3: Normalized spectra of 3x3 mesh for actual and $M + 1$ th root approximated boundary for range of $\mathbf{a} = [0.5, 1]$

be directly applied to the numeric coefficients of each series second order section generated from analysis of the mesh for $\mathbf{a} = 1$.

$$\begin{aligned}
 H(z) &= g \prod_{i=1}^N \frac{b_0 - 2 \operatorname{Re}(f(\mathbf{a})q_i)z^{-1} + |f(\mathbf{a})q_i|^2 z^{-2}}{1 - 2 \operatorname{Re}(f(\mathbf{a})p_j)z^{-1} + |f(\mathbf{a})p_j|^2 z^{-2}} \\
 &= g \prod_{i=1}^N \frac{b_0 - 2f(\mathbf{a}) \operatorname{Re}(q_i)z^{-1} + f(\mathbf{a})^2 |q_i|^2 z^{-2}}{1 - 2f(\mathbf{a}) \operatorname{Re}(p_j)z^{-1} + f(\mathbf{a})^2 |p_j|^2 z^{-2}} \\
 &= g \prod_{i=1}^N \frac{b_0 + f(\mathbf{a})b_{i,1}z^{-1} + f(\mathbf{a})^2 b_{i,2}z^{-2}}{1 + f(\mathbf{a})a_{i,1}z^{-1} + f(\mathbf{a})^2 a_{i,2}z^{-2}}
 \end{aligned} \tag{5.3}$$

Empirically, it was found that the parallel second order representation can also implement an approximated boundary parameter in the same fashion as equation (5.3) and produce equivalent results to those of figure 5.3.

$$H(z) = g \sum_{i=1}^N \frac{b_0 + f(\mathbf{a})b_{i,1}z^{-1}}{1 + f(\mathbf{a})a_{i,1}z^{-1} + f(\mathbf{a})^2 a_{i,2}z^{-2}} \tag{5.4}$$

5.2 Trajectory of Zeros and Poles

Each series second order section can also be expressed in terms of the zero and pole radii and angles. Such a representation allows control over both the radius and angle of each pole and zero according to some function.

$$H(z) = g \prod_{i=1}^N \frac{b_0 - 2r_i \cos \theta_i z^{-1} + r_i^2 z^{-2}}{1 - 2r_j \cos \theta_j z^{-1} + r_j^2 z^{-2}} \quad (5.5)$$

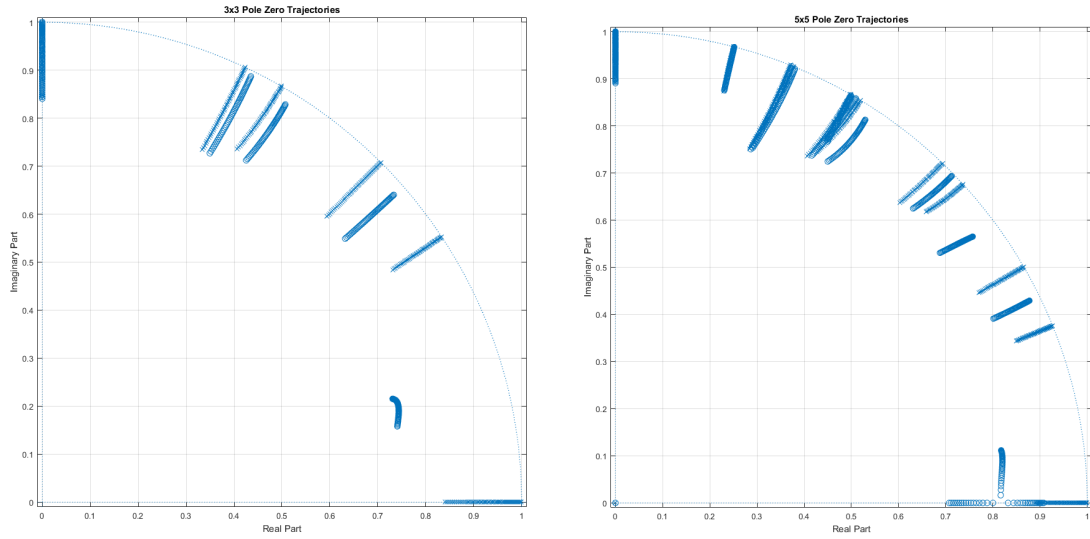


Figure 5.4: First quadrant trajectories of zeros and poles for 3x3 and 5x5 over $a = [0.5, 1]$. Other quadrants are symmetric.

Inspection of the trajectories and curves in figure 5.4 and 5.5 reveals more details about the behavior of the poles and zeros for varying $\mathbf{a}$. As it turns out, the angles of zeros and poles do vary somewhat, but, for all except one obvious zero in figure 5.4, they do so at such a small scale that the original assumption of a constant angle is still tenable.

5.2.1 Polynomial Approximation

For an improved approximation over figure 5.3, polynomials for each zero and pole can be fit to the radius curves in figure 5.5 (using `polyfit` in Matlab), and

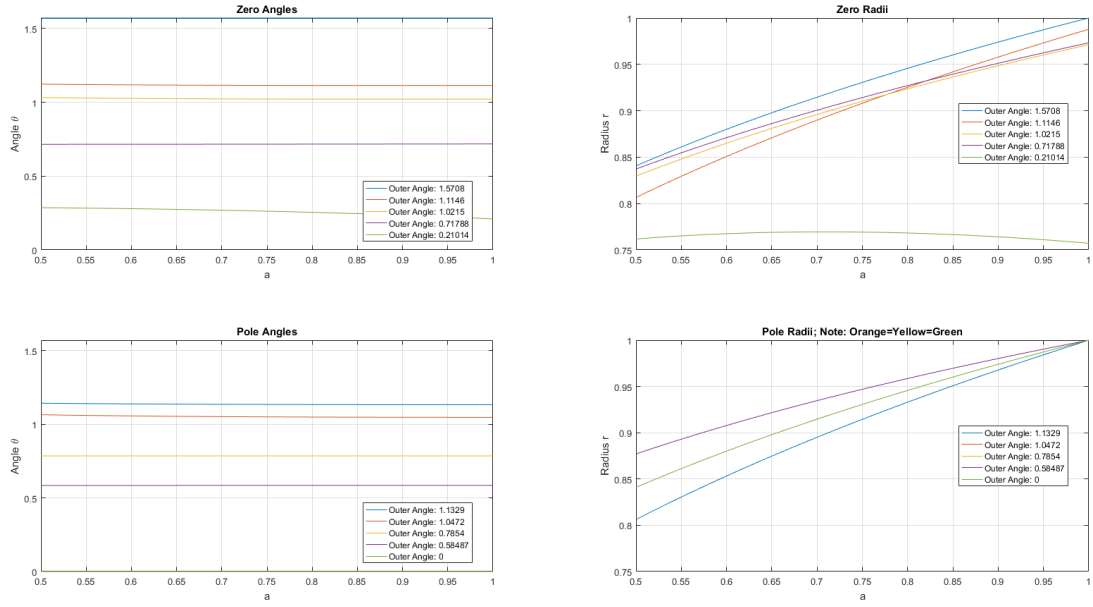


Figure 5.5: 3x3 angle and radius curves for the first quadrant. Outer angle corresponds to angle of zero or pole at $\mathbf{a} = 1$

then used to construct second order sections of the form in equation (5.5), with constant θ . The results of this method using a 5th order polynomial can be seen in figure 5.6. Smaller order polynomials, such as quadratic, also produce a frequency response with minimal error, but result in significantly shorter amplitude envelopes compared to those of the original boundary conditions (for example, the $\mathbf{a} = 1$ case has a decaying envelope for quadratic approximation). It is also important to verify that the approximation polynomial does not exceed 1 in the marginally stable case ($\mathbf{a} = 1$) in order to avoid instability.

5.2.2 Zero Pole Tracking

In order to automatically generate the curves in figure 5.5 necessary for polynomial approximation, the zeros and poles are sorted according to angle around the unit circle. Unfortunately, for cases such as the 5x5 in figure 5.7 and larger dimensions, the trajectories of zero and poles cross over one another, breaking down the ability to track specific poles and zeros over changing $\mathbf{a}$ simply by sorting their angles. In order to extend the polynomial approximation method to larger meshes,

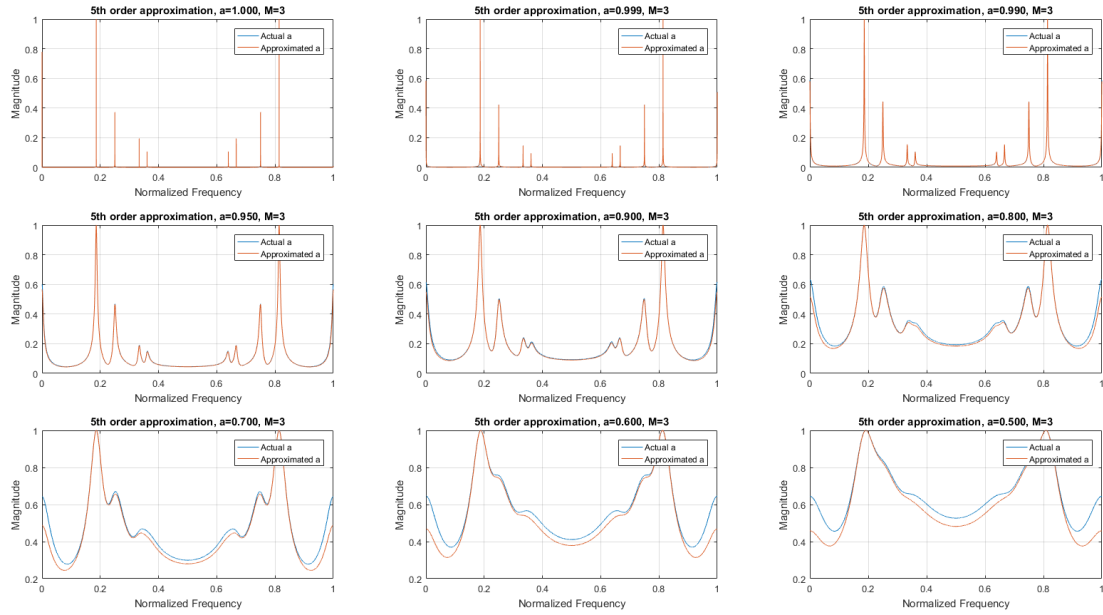


Figure 5.6: Normalized spectra of 3x3 mesh for actual and polynomial approximated boundary for range of $\mathbf{a} = [0.5, 1]$

a more robust technique to track zero and pole trajectories that potentially cross one another must be developed.

5.3 Approximating Excitation Location

The parallel second order section representation is ideal for approximating the behavior of changing excitation location. Given that each mode vibrational mode has an associated filter, each biquad in the parallel bank can be amplified or attenuated independently. Inspection of the overlaid zero and pole plots of a parallel bank of second order sections in figure 5.8 reveals that each section can be expressed in terms of a pole and a unique gain equal to $2r_i$. This factored version of equation (4.13) works because each residue r_i is always real.

$$\tilde{H}(z) = \sum_{i=1}^M 2r_i \frac{1 - \text{Re}(p_i)z^{-1}}{1 - 2\text{Re}(p_i)z^{-1} + |p_i|^2 z^{-2}} \quad (5.6)$$

Using the knowledge from section 4.2.1 that all excitation points on a mesh

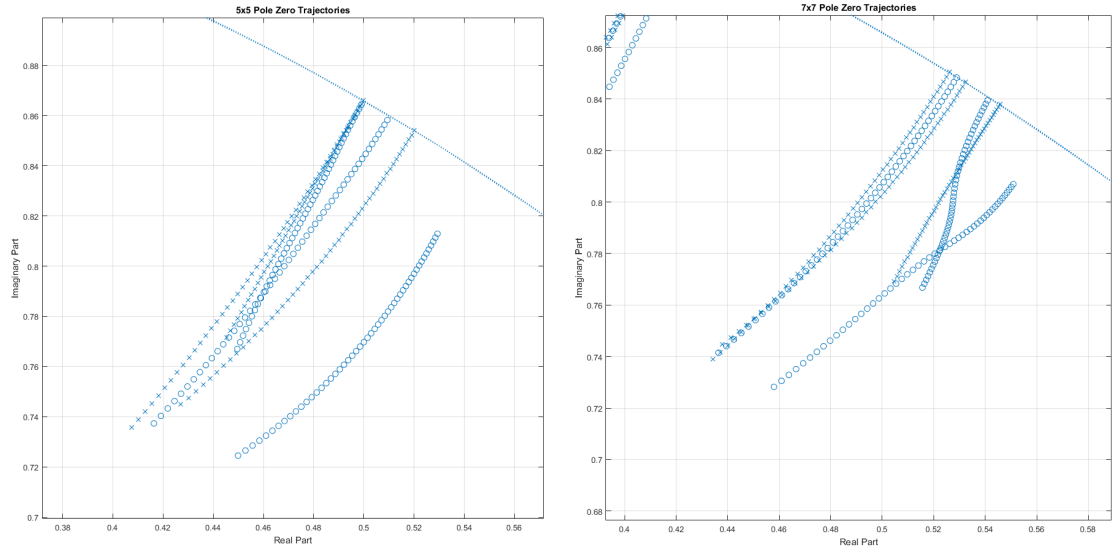


Figure 5.7: Magnified quadrant trajectories of zeros and poles where crossover occurs for 5x5 and 7x7 over $a = [0.5, 1]$. Other quadrants are symmetric.

of size M have common poles, the frequency responses at various excitation points (figure 5.9) are understood to have common peaks with varying magnitudes. This observation motivates the intuition that a maximal set of parallel second order sections could be constructed from a few excitation points, such as the corner, center, and middle edge. Modeling the selection of excitation point is simply a matter of mixing (attenuating and amplifying) sections. The gains of modes that are not present, such as in the center excitation, would of course be 0.

Based on figure 5.9, a more central excitation amplifies midrange modes closer to $\frac{fs}{4}$ (where fs is the sample rate), and edge excitations emphasizes the lower and higher frequency modes. One such approach to mixing, as seen in figure 5.10, would be to store three vectors of associated gains $g_i = 2 \operatorname{Re}(r_i)$ for each excitation point (corner, center and edge), and interpolate between them based on the geometric position of the desired excitation point on the mesh. This method does fail to accurately recreate the behavior of a few modes that are not present in the corner, edge and center points, but still provides a useful parametric control to control timbre. There is certainly potential for improved mixing or interpolation strategies as well.

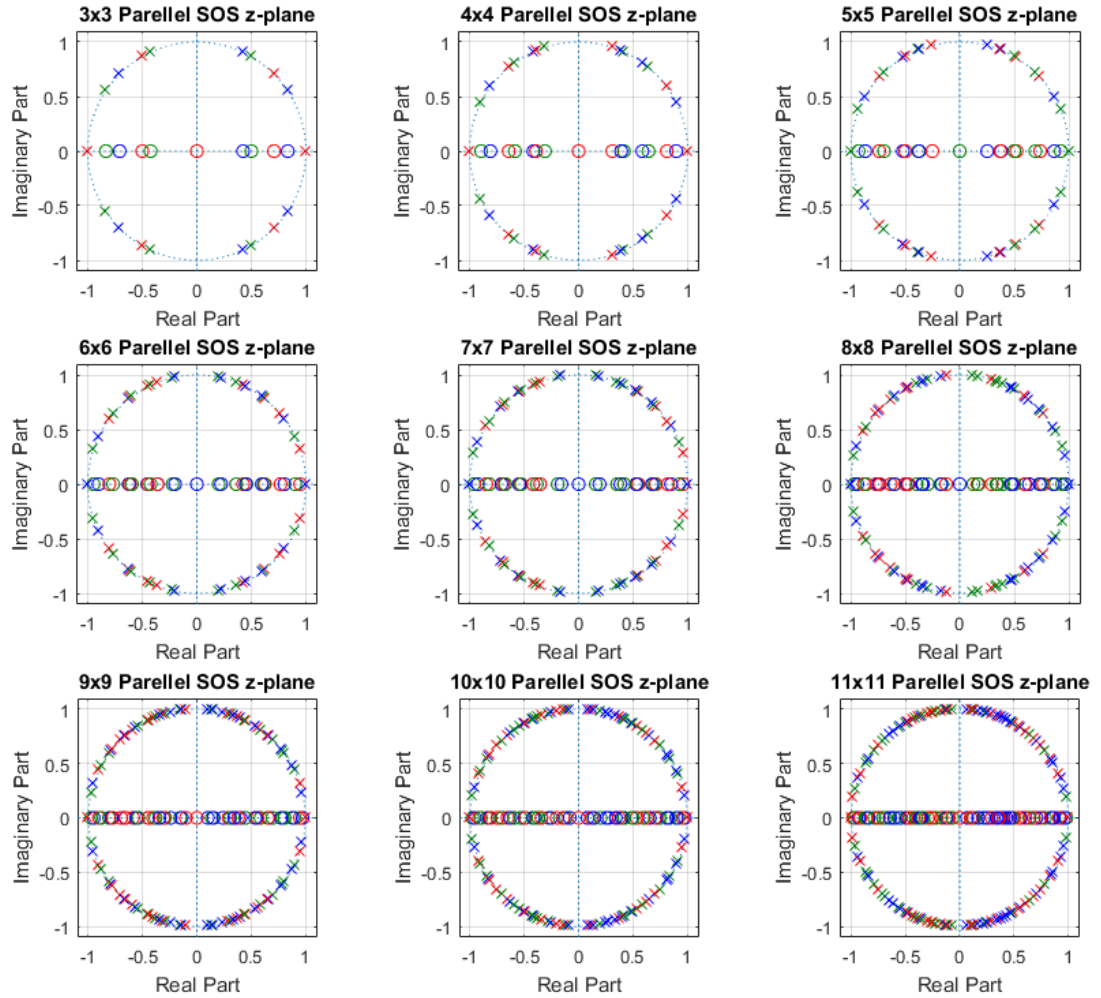


Figure 5.8: Zero pole plot of parallel second order sections, colors represent vertically paired complex conjugate poles and zero that form a section.

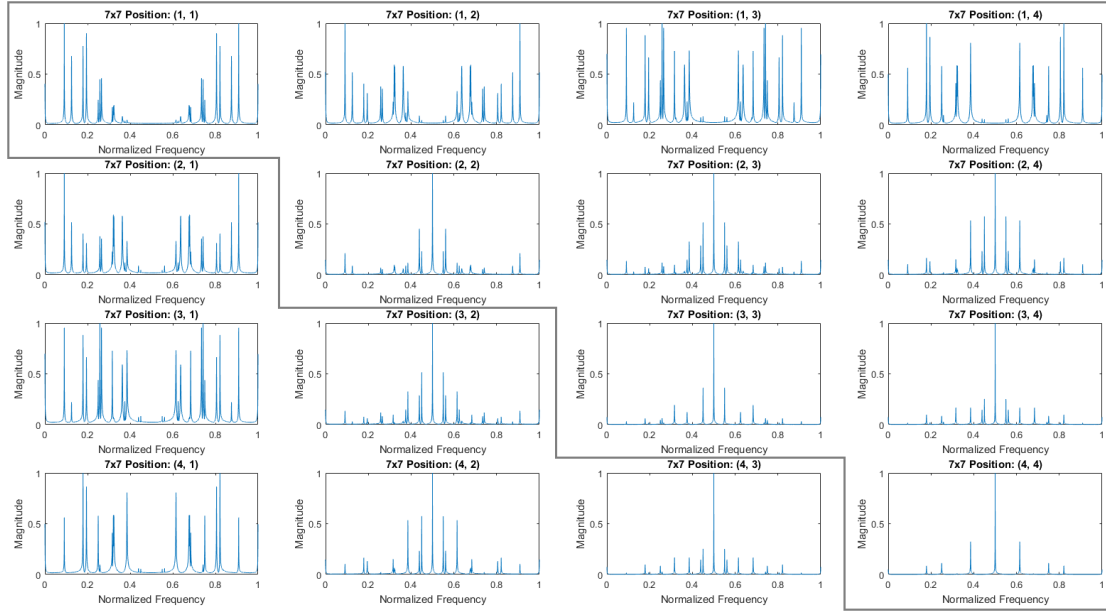


Figure 5.9: Frequency response of various excitations points in the symmetric upper left quadrant of 7x7 mesh. Outline specifies unique possible excitations.

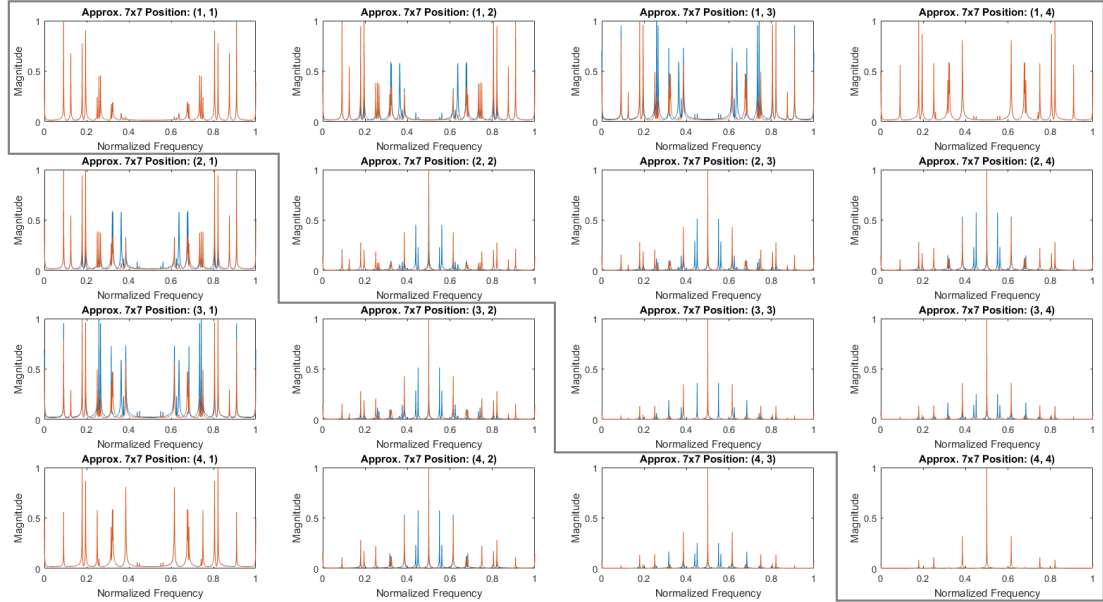


Figure 5.10: Frequency response actual (blue) and approximated (orange) excitations points in the symmetric upper left quadrant of 7x7 mesh. Outline specifies unique possible excitations.

Chapter 6

Conclusion

6.1 Transfer Function of the Mesh

While the symbolic analysis that derived the parametric transfer function is an interesting result to inspect the mesh with, the insurmountable computational expense of symbolic computing along with quantization issues that arise when attempting to implement the filter in a fixed precision environment limit its practical uses for sound synthesis.

6.2 Parametric Instrument Design

The results of the work presented in this thesis is enough to begin to piece together a real-time, parametric, mesh-sounding instrument that is more conducive to real-time synthesis.

- Construct a state space model of the mesh, selecting mesh size, excitation position and boundary reflection behavior.
- Extract the zero pole representation from the state space model using eigenanalysis, and cancel equivalent poles and zeros.
- Return to the minimal state space model, and extract parallel second order sections from the diagonalized state space representation.

- Repeat the previous steps for a few excitation positions to build a maximal set of biquad filter 'modes' to mix between to approximate excitation location.
- Reintroduce parameters to the coefficients of the second order sections to approximate the behavior of varying boundary conditions.

Since the 2D Waveguide Mesh model is itself an approximation of a physical system, it seems unnecessary to hold the filter based representation to the expectation that it perfectly recreate the behavior of the mesh. The parametric instrument's ability to recreate sounds that are perceptually similar or nearly identical to the original physical model is a sufficient criteria for success. The approximated parameter behavior performs well enough, but the visible error in figures 5.3 and 5.10 indicate that improved approximations can certainly be developed.

6.2.1 Boundary Filters

Though not detailed in chapters 4 and 5, it is possible to embed the states of boundary filters, as opposed to scalars, into the state space model, and make similar observations about how the trajectory of resulting zeros and poles change with respect to the boundary filter parameters (such as frequency cutoff). Since boundary filters such as the one pole low pass affect the angle of zeros and poles significantly, polynomial approximations of both the angle and radius would be required. The issues raised in section 5.2.2 are still outstanding; improved methods for tracking the trajectories of poles and zeros with respect to the boundary parameter need to be developed in order to perform useful polynomial approximation for any sized mesh.

6.3 Modal Representation

By converting the 2D waveguide mesh into a filter, it's computation is effectively being reduced from an M^2 grid to only the mesh's resonant modes. The consistent quantity of pole zero cancellation that occurs in every mesh analysis serves as strong evidence to this observation. The bank of second order sections

that implements the mesh can then be considered the modal representation of the mesh ([29]). The parallel second order section especially so, as existing research ([30]) has attempted to formulate a very similar pole residue modal representation of a room transfer function (in 3D) using different methods.

The parallel second order section formulation (5.6) is also unconstrained by the physical model representation. In this form, it is possible to selectively remove undesirable modes such as the peak at $\frac{fs}{2}$ caused by central excitation positions (as seen in figures 5.9 and 5.10), or at DC. The mixture of each mode (the vector of g_i) can also be designed arbitrarily as well to create timbre not immediately realizable by the physical model. With this approach, the minimal state space model (section 4.4) need not be utilized, as the gain-less parallel second order sections can simply be constructed from the poles of a singular excitation position (such as the corner), and their individual gains chosen creatively.

6.4 Complexity

Indeed, it is worth reiterating that the uncanceled transfer function requires the same amount of filter state memory (z^{-1}) as the original model, so while a particular filter order might perform better than it's physical model counterpart, the complexity of the uncanceled transfer function still grows quadratically with respect to the mesh dimension M . A reasonable assertion might be to quantify the complexity of the 'modal' representation according to it's highest possible filter order that can be generated: the corner (at least for equivalent I/O location). In this case, the resulting complexity of the canceled transfer function is consistently less than half the uncanceled complexity of $4M^2 + 4M$.

Appendix A

Waveguide Difference Equations for Border Cases

Top left corner:

$$\begin{aligned} N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - aN_{i,j}(n-2) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\ S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + aN_{i,j}(n-2) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\ E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) + aW_{i,j}(n-2) - W_{i,j+1}(n-1)) \\ W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) - aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \end{aligned} \tag{A.1}$$

Top row:

$$\begin{aligned} N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - aN_{i,j}(n-2) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\ S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + aN_{i,j}(n-2) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\ E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) + E_{i,j-1}(n-1) - W_{i,j+1}(n-1)) \\ W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) - E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \end{aligned} \tag{A.2}$$

Top right corner:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - aN_{i,j}(n-2) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + aN_{i,j}(n-2) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) + E_{i,j-1}(n-1) - aE_{i,j}(n-2)) \\
W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + aN_{i,j}(n-2) - E_{i,j-1}(n-1) + aE_{i,j}(n-2))
\end{aligned} \tag{A.3}$$

Left column:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - S_{i-1,j}(n-1) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\
S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\
E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + aW_{i,j}(n-2) - W_{i,j+1}(n-1)) \\
W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) - aW_{i,j}(n-2) + W_{i,j+1}(n-1))
\end{aligned} \tag{A.4}$$

Right column:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) - S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
S_{i,j}(n) &= \frac{1}{2}(-N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
E_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) - aE_{i,j}(n-2)) \\
W_{i,j}(n) &= \frac{1}{2}(N_{i+1,j}(n-1) + S_{i-1,j}(n-1) - E_{i,j-1}(n-1) + aE_{i,j}(n-2))
\end{aligned} \tag{A.5}$$

Bottom left corner:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) - S_{i-1,j}(n-1) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\
S_{i,j}(n) &= \frac{1}{2}(-aS_{i,j}(n-2) + S_{i-1,j}(n-1) + aW_{i,j}(n-2) + W_{i,j+1}(n-1)) \\
E_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) + aW_{i,j}(n-2) - W_{i,j+1}(n-1)) \\
W_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) - aW_{i,j}(n-2) + W_{i,j+1}(n-1))
\end{aligned} \tag{A.6}$$

Bottom row:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) - S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\
S_{i,j}(n) &= \frac{1}{2}(-aS_{i,j}(n-2) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + W_{i,j+1}(n-1)) \\
E_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) - W_{i,j+1}(n-1)) \\
W_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) - E_{i,j-1}(n-1) + W_{i,j+1}(n-1))
\end{aligned} \tag{A.7}$$

Bottom right corner:

$$\begin{aligned}
N_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) - S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
S_{i,j}(n) &= \frac{1}{2}(-aS_{i,j}(n-2) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) + aE_{i,j}(n-2)) \\
E_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) + E_{i,j-1}(n-1) - aE_{i,j}(n-2)) \\
W_{i,j}(n) &= \frac{1}{2}(aS_{i,j}(n-2) + S_{i-1,j}(n-1) - E_{i,j-1}(n-1) + aE_{i,j}(n-2))
\end{aligned} \tag{A.8}$$

Bibliography

- [1] Gary P. Scavone, “Simulations, munt 618: Computational modeling of musical acoustic systems. week 12,” 2016.
- [2] Tamara Smyth, Jennifer Hsu, and Ryan Done, “Toward a real-time waveguide mesh implementation,” in *Proceedings of the International Symposium on Musical Acoustics*. International Symposium on Musical Acoustics, forthcoming 2017.
- [3] S Van Duyne and Julius O Smith, “The 2-d digital waveguide mesh,” in *Applications of Signal Processing to Audio and Acoustics, 1993. Final Program and Paper Summaries., 1993 IEEE Workshop on*. IEEE, 1993, pp. 177–180.
- [4] Damian T Murphy, Chris JC Newton, and David M Howard, “Digital waveguide mesh modelling of room acoustics: Surround-sound, boundaries and plugin implementation,” in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx)*, 2001.
- [5] Marco Novello, Federico Fontana, and Enrico Bozzo, “Decimation of finite-difference time-domain schemes in 1d and 2d boundary-absorbing acoustic model simulations,” in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 8227–8231.
- [6] Maarten Van Walstijn and Konrad Kowalczyk, “On the numerical solution of the 2d wave equation with compact fdtd schemes,” *Proc. Digital Audio Effects (DAFx), Espoo, Finland*, pp. 205–212, 2008.
- [7] Stefan Bilbao, “Modeling of complex geometries and boundary conditions in finite difference/finite volume time domain room acoustics simulation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 7, pp. 1524–1533, 2013.
- [8] Stefan Bilbao and Craig J Webb, “Physical modeling of timpani drums in 3d on gpgpus,” *Journal of the Audio Engineering Society*, vol. 61, no. 10, pp. 737–748, 2013.

- [9] Craig J Webb and Stefan Bilbao, “Computing room acoustics with cuda-3d fdtd schemes with boundary losses and viscosity,” in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*. IEEE, 2011, pp. 317–320.
- [10] Julius O. Smith, *Physical Audio Signal Processing*, <http://ccrma.stanford.edu/~jos/pasp/>, accessed 2017-05-01, online book, 2010 edition.
- [11] Matti Karjalainen and Cumhur Erkut, “Digital waveguides versus finite difference structures: Equivalence and mixed modeling,” *EURASIP Journal on Applied Signal Processing*, vol. 2004, pp. 978–989, 2004.
- [12] Mark J Beeson and Damian T Murphy, “Roomweaver: A digital waveguide mesh based room acoustics research tool,” in *Proc. 7th Int. Conf. Digital Audio Effects (DAFx-04)*, 2004, pp. 268–273.
- [13] Ravish Mehra, Nikunj Raghuvanshi, Lauri Savioja, Ming C Lin, and Dinesh Manocha, “An efficient gpu-based time domain solver for the acoustic wave equation,” *Applied Acoustics*, vol. 73, no. 2, pp. 83–94, 2012.
- [14] Stefan Bilbao, Brian Hamilton, Alberto Torin, Craig Webb, Paul Graham, Alan Gray, Kostas Kavoussanakis, and James Perry, “Large scale physical modeling sound synthesis,” in *Proceedings of the Stockholm music acoustic conference (SMAC2013)*, Stockholm, 2013, pp. 593–600.
- [15] Craig J Webb and Stefan Bilbao, “On the limits of real-time physical modelling synthesis with a modular environment,” .
- [16] Finite-Difference Time-Domain Schemes, “Decimation in time and space of finite-difference time-domain schemes: Standard isotropic lossless model,” 2015.
- [17] Zhixin Chen and Robert C Maher, “Analytical expression for impulse response between two nodes in 2-d rectangular digital waveguide mesh,” *IEEE Signal Processing Letters*, vol. 15, pp. 221–224, 2008.
- [18] Scott A Van Duyne and Julius O Smith, “The tetrahedral digital waveguide mesh,” in *Applications of Signal Processing to Audio and Acoustics, 1995., IEEE ASSP Workshop on*. IEEE, 1995, pp. 234–237.
- [19] Damian T Murphy and David M Howard, “2-d digital waveguide mesh topologies in room acoustics modelling,” in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx)*, 2000, pp. 211–216.

- [20] Guilherme Campos and David Howard, “A parallel 3d digital waveguide mesh model with tetrahedral topology for room acoustic simulation,” in *Proceedings of the COST G-6 Conference on Digital Audio Effects (DAFx), (Verona, Italy)*, 2000, pp. 73–78.
- [21] Perry R. Cook and Gary P. Scavone, “The synthesis toolkit in c++ (stk),” 2016.
- [22] IEEE Standards Committee et al., “754-2008 ieee standard for floating-point arithmetic,” *IEEE Computer Society Std*, vol. 2008, 2008.
- [23] Alan V. Oppenheim, Ronald W. Schaffer, and John R. Buck, *Discrete-time Signal Processing (2Nd Ed.)*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1999.
- [24] Julius O. Smith, *Introduction to Digital Filters with Audio Applications*, <http://ccrma.stanford.edu/~jos/filters/>, accessed 2017-05-01, online book.
- [25] Emilio Frazzoli and Munther Dahleh, “Dynamic systems and control, spring 2011 (6.241j),” *Massachusetts Institute of Technology: MIT OpenCourseWare*, <https://ocw.mit.edu> (accessed 2017-05-01), Creative Commons BY-NC-SA.
- [26] Howard Harry ROSENBROCK, “Structural properties of linear dynamical systems,” *International Journal of Control*, vol. 20, no. 2, pp. 191–202, 1974.
- [27] HH Rosenbrock, “The zeros of a system,” *International Journal of Control*, vol. 18, no. 2, pp. 297–299, 1973.
- [28] Alan J Laub and BC Moore, “Calculation of transmission zeros using qz techniques,” *Automatica*, vol. 14, no. 6, pp. 557–566, 1978.
- [29] Perry R Cook, “Physically informed sonic modeling (phism): Synthesis of percussive sounds,” *Computer Music Journal*, vol. 21, no. 3, pp. 38–49, 1997.
- [30] Yoichi Haneda, Yutaka Kaneda, and Nobuhiko Kitawaki, “Common-acoustical-pole and residue model and its application to spatial interpolation and extrapolation of a room transfer function,” *IEEE Transactions on Speech and Audio Processing*, vol. 7, no. 6, pp. 709–717, 1999.