

## ABSTRACT

Title of Dissertation: REAL-TIME AUDIO REVERBERATION  
FOR VIRTUAL ROOM ACOUSTICS

Justin M Shen  
Master of Science, 2020

Dissertation Directed by: Professor Ramani Duraiswami  
Department of Computer Science

For virtual and augmented reality applications, it is desirable to render audio sources in the space the user is in, in real-time without sacrificing the perceptual quality of the sound. One aspect of the rendering that is perceptually important for a listener is the late-reverberation, or “echo”, of the sound within a room environment. A popular method of generating a plausible late reverberation in real-time is the use of Feedback Delay Network (FDN). However, its use has the drawback that it first has to be tuned (usually manually) for a particular room before the late-reverberation generated becomes perceptually accurate. In this thesis, we propose a data-driven approach to automatically generate a pre-tuned FDN for any given room described by a set of room parameters. When combined with existing method for rendering the direct path and early reflections of a sound source, we demonstrate the feasibility of being able to render audio source in real-time for interactive applications.

REAL-TIME AUDIO REVERBERATION  
FOR VIRTUAL ROOM ACOUSTICS

by

Justin M Shen

Dissertation submitted to the Faculty of the Graduate School of the  
University of Maryland, College Park in partial fulfillment  
of the requirements for the degree of Master of Science 2020

Advisory Committee:  
Professor Ramani Duraiswami, Chair/Advisor  
Professor Matthias Zwicker  
Assistant Professor Nirupam Roy

© Copyright by  
Justin M Shen  
2020

## Acknowledgments

I would like to take some time to acknowledge all the people who have made an impact in my graduate experience so far through the various interactions we have had. Without them, my graduate experience simply would not be the same, and I am grateful for all their encouragements and advice throughout the thesis process.

First, I would like to thank my advisor, Professor Ramani Duraiswami, for many years of guidance and teachings. Even as far back as my high school years, he was open to introducing me to the world of scientific research by providing me with the opportunity to work in his lab for my high school senior research project. The opportunity played no small part in my eventual decision to study Computer Science in college. My education throughout my undergraduate and graduate years has been greatly enriched thanks to his presence and encouragements.

I would also like to thank my committee members Professor Matthias Zwicker and Nirupam Roy. They have both been teachers to courses I've had in the past whose teachings helped expand my horizon and proved to be useful in my research. I am truly grateful for their flexibility and willingness to be a part of my committee, especially during this unusual time of COVID-19.

Lastly, I would like to thank everyone else who made my graduate experience possible: all my friends and family who have cared and encouraged me along the way. Mr. Jason Filippou, Professor David Jacobs, Dr. Ilchul Yoon, Professor Updaya Shankar, and all other professors who have provided me the opportunity to work as a teaching assistant throughout my undergraduate and graduate years. Tom

Hurst for his stellar administrative support to graduate students. Dorothea Brosius and the Institute for Research in Electronics and Applied Physics for providing the LaTeX thesis template. Once again, I owe my gratitude to all the people who have made this thesis possible and have made my graduate experience unique.

## Table of Contents

|   |      |
|---|------|
| Acknowledgements  | ii   |
| Table of Contents   | iv   |
| List of Tables  | vi   |
| List of Figures   | vii  |
| List of Abbreviations   | viii |
| 1 Introduction  | 1    |
| 2 Background  | 4    |
| 2.1 Artificial Reverberation . . . . .                        | 4    |
| 2.2 Convolutional and Computational Acoustics . . . . .       | 6    |
| 2.3 Delay Network and Feedback Delay Network . . . . .        | 7    |
| 2.3.1 Feedback Matrix . . . . .                               | 8    |
| 2.3.2 Delay Line and Length . . . . .                         | 9    |
| 2.4 Important Perceptual Metrics for Reverberation . . . . .  | 10   |
| 2.5 Automatic Tuning of Feedback Delay Network . . . . .      | 12   |
| 3 Methodology   | 14   |
| 3.1 Automatic FDN Construction from Room Parameters . . . . . | 14   |
| 3.2 Obtaining the Room Impulse Response . . . . .             | 15   |
| 3.3 Automatic Design of FDN . . . . .                         | 16   |
| 3.4 Choice of Perceptual Metric . . . . .                     | 17   |
| 3.5 SVM Regression for Generating FDN Parameters . . . . .    | 18   |
| 4 Results   | 20   |
| 4.1 Genetic Algorithm Optimization Loss Value . . . . .       | 20   |
| 4.2 SVM Results . . . . .                                     | 21   |
| 4.2.1 Small Room Data Set Results . . . . .                   | 22   |
| 4.2.2 Large Room Data Set Results . . . . .                   | 25   |
| 4.3 Real-time System Demonstration . . . . .                  | 26   |

|     |   |    |
|-----|---|----|
| 5   | Discussions                                 | 28 |
| 5.1 | Quality of the Training Data Set . . . . .  | 28 |
| 5.2 | SVM Regression Performance . . . . .        | 30 |
| 5.3 | Computational Cost Considerations . . . . . | 31 |
| 6   | Conclusion                                  | 33 |
| 6.1 | Summary . . . . .                           | 33 |
| 6.2 | Possible Future Work . . . . .              | 34 |

## List of Tables

|     |   |    |
|-----|---|----|
| 4.1 | Training performance for SVM regression model for FDN parameterization ( $g, m_1, m_2, m_3, m_4$ ) for the small room data set. . . . . | 23 |
| 4.2 | Testing performance for SVM regression model for FDN parameterization ( $g, m_1, m_2, m_3, m_4$ ) for the small room data set. . . . .  | 23 |
| 4.3 | Training performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the small room data set. . . . .               | 24 |
| 4.4 | Testing performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the small room data set. . . . .                | 25 |
| 4.5 | Training performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the large room data set. . . . .               | 26 |
| 4.6 | Testing performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the large room data set. . . . .                | 26 |



## List of Figures

|     |  |    |
|-----|--|----|
| 2.1 | Anatomy of a typical room impulse response from [7] . . . . .  | 5  |
| 2.2 | Example FDN structure from [7] . . . . .   | 8  |
| 2.3 | Example of a simulated impulse response from a FDN matched to have the same energy as the measured IR [11]. . . . .  | 13 |
| 3.1 | Flow chart of the proposed general method. The final regression model to deploy for inference is boxed in red. . . . .   | 15 |
| 4.1 | Spectrogram of sound signal convolved with RIR and FDN produced RIR for various loss values. In each set of three spectrogram, the top is the original audio, middle is the result produced from the simulated RIR and bottom is the result produced from FDN generated RIR. . .   | 21 |
| 4.2 | Sample output from the SVM regression model trained on the small room data set with FDN parameterized as $(g, m_1, m_2, m_3, m_4)$ . In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom). . . . . | 24 |
| 4.3 | Sample output from the SVM regression model trained on the small room data set with FDN parameterized as $(g, b, c)$ . In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom). . . . .               | 25 |
| 4.4 | Sample output from the SVM regression model trained on the large room data set with FDN parameterized as $(g, b, c)$ . In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom). . . . .               | 26 |
| 4.5 | Plot of the total amount of time the mock demo took to process different number of audio samples. . . . .  | 27 |

## List of Abbreviations

|      |                                     |
|------|-------------------------------------|
| BEM  | Boundary Element Method             |
| BRIR | Binaural Room Impulse Response      |
| EDC  | Energy Decay Curve                  |
| FDN  | Feedback Delay Network              |
| FDTD | Finite-Difference Time-Domain       |
| FEM  | Finite Element Method               |
| FFT  | Fast Fourier Transform              |
| HRTF | Head-Related Transfer Function      |
| IR   | Impulse Response                    |
| MFCC | Mel-frequency cepstral coefficients |
| RIR  | Room Impulse Response               |
| SVM  | Support Vector Machine              |

## Chapter 1: Introduction

When a sound wave is propagating through a room, what is perceived by a listener is typically composed of the direct signal from the sound source to the listener, as well as the various indirect signals from the sound source bouncing off of the surfaces of the room. This perceived *reverberated* sound can be divided into two perceptually different segments based on the room impulse response (RIR). The first segment is referred to as the *early reflection*, while the last segment is referred to as the *late reverberation* [19]. In this thesis, we are primarily interested in developing a framework for efficient approximation of the late reverberation of audio signals in an arbitrary environment.

Being able to approximate the late reverberation in real-time is important for interactive applications where realistic audio rendering is required. For example, in augmented and virtual reality applications, it is often desirable to create as immersive or realistic of an experience as possible. This can be realized with a traditional head up display hardware which can render visual elements and play audio. In particular, we can imagine augmented or virtual reality use case such as virtual concert performance where realistic audio can dramatically enhance the listening experience since music being perform in an amphitheater or concert hall carries very rich tone.

Similarly, for virtual conference meetings and virtual classrooms, realistic audio and visuals can potentially provide a higher level of engagement and more closely replicate the experience of in-person meetings, which is especially useful for times when in-person meetings might be inconvenient.

In audio rendering applications, the original, unaltered audio signal that a listener will hear is referred to as the *dry audio*, and the process of the dry audio propagating through a room environment and bouncing off of the surfaces of the room to form the final reverberated sound a listener perceives can be referred to as *reverberation*. The point-to-point RIR of a room characterizes the behavior of a sound traveling from a source location to a receiver location, and the convolution of the dry audio signal with the RIR can be used to accurately render the reverberated sound [10]. Thus, to render audio that is similar to how a sound is actually perceived by a listener in a given room, it is often sufficient to compute the RIR of a room, along with the Head-Related Transfer Function (HRTF) unique to an individual. While we can accurately compute a reverberated sound through convolution, that alone is not enough if we are interested in being able to render the reverberated sound in real-time [18] [19].

To render the reverberated sound in real-time, one approach is to approximate the late-reverberation using a Feedback Delayed Network (FDN), allowing us to apply other existing methods for rendering just the direct path and early reflections of the sound rather than the entire sound source [19]. This approach, however, has the drawback that the FDN must be first tuned for a particular room (often manually) before the late-reverberation generated becomes perceptually accurate.

To overcome this drawback, we propose a data-driven approach to automatically generate a pre-tuned FDN for any given room described by a set of room parameters. This approach involves building a data set to train a model to learn a mapping between a parametric model of a room and the parameteric model of the FDN corresponding to those rooms. The main requirement for this model is that once the model is built, it can be use to infer the FDN parameters in real-time once information about the room environment is known. This, when combined with existing methods for rendering the direct path and early reflections of a sound source, allows us to render more realistic audio in real-time.

In the next section we will discuss relevant background and previous works related to our proposed approach. After the background, we will then discuss the methodology of our approach and the results of our implementation in sections 3 and 4. Finally, we provide some discussions and conclusion regarding our work in the remaining sections.

## Chapter 2: Background

### 2.1 Artificial Reverberation

As mentioned earlier, reverberation is the result of sound bouncing off the surfaces of the room. More specifically, as the sound propagates through an environment, it is slowed down from interaction with the environment, giving the listener a sense of the space and structure of the environment. Furthermore, a sound source can bounce off surfaces in the room multiple times, meaning the resulting sound the listener hears is a combination of various paths the sound waves took prior to reaching the ears.

Sound that have only bounces off surfaces a couple of times can be heard more distinctly are referred to as the early reflections of the sound. While the sound that traveled directly to the listener (the direct path) allows listeners to perceive the direction of the sound, the early reflections are distinct enough to give listeners a sense of the geometry and material of the room [19]. The early reflection ends once the reverberation reaches its “asymptotic statistical behavior,” but it is typically taken to be the first 80 to 100 ms of the sound signal [13]. An example of the different parts of a room impulse response (RIR) for characterizing the interaction of the sound with the environment is shown in Fig. 2.1. Since the direct path and

early reflections are rather distinct, it is important to be able to accurately simulate them in order to render sound realistically. In contrast, the late reverberation that resulted from the sound bouncing off the surfaces of the room many times becomes less distinct and conveys a sense of the size of the environment and its absorbing power. Furthermore, this late reverberation can be characterized statistically, making it acceptable to approximate it without significant loss in realism [19].

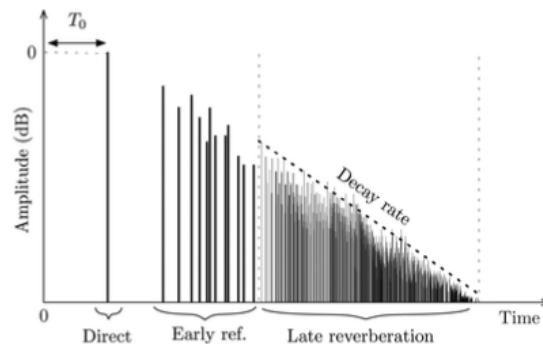


Figure 2.1: Anatomy of a typical room impulse response from [7]

Since the idea of recreating reverberation through artificial reverberation was first introduced by Schroeder in 1961, three main types of algorithms have been developed: delay networks, convolutional, and computational acoustic [19]. Convolutional approaches focus on obtaining good physical measurements of a room and being able to compute the convolution efficiently. Computational acoustic approaches simulate the propagation of acoustic signals in a geometric representation of the room and can be further broken down into two variants: one being the wave-based methods that aim to solve the wave equation numerically, and the other being ray-based methods that propagate sound as rays geometrically. Delay network approaches use networks of delay lines and digital filters to recreate the sound delays of a reverberation. A popular example of a delay networks structure is the

Feedback Delay Networks (FDNs), which will be the main focus of this thesis for producing reverberation in real-time.

## 2.2 Convolutional and Computational Acoustics

The goal of convolutional approaches and computational acoustics approaches for simulating artificial reverberation is to compute the room impulse response (RIR). Once the RIR is computed the artificial reverberation can be obtained by finding the convolution of the dry audio signal and the computed RIR [10] [18]. Both approaches are both physically based, allowing them to accurately reproduce the reverberated audio.

As mentioned previously, convolutional approaches are concerned with physically measuring the RIR through sound recordings and applying the measured RIR to generate artificial reverberation efficiently. Once a recording of the RIR is obtained, post-processing algorithm is necessary to deal with noise and other limitations related to the recording device [19]. With the post-processing complete, the Fast Fourier Transform (FFT) algorithm and its variants are used to efficiently compute the convolution between the dry audio signal and the impulse response.

Meanwhile, for wave-based computational acoustics approaches, the wave equation can be solved through various numerical techniques in either the time or frequency domain [19]. Notable time domain techniques include the finite-difference time-domain (FDTD) technique and its variants [5] [19]. For the frequency domain, finite element method (FEM) [15] and boundary element method (BEM) are used



[4] [9] [19].

Wave-based approaches are considered to be the most accurate way to simulate the RIR, however, they are also very computationally expensive. Ray-based computational acoustics approaches (also called geometrical acoustics) offers alternative techniques that are faster but less accurate [17] [19]. Because ray-based approaches treat sound as rays, it is more accurate for mid to high frequency sound waves and has difficulty capturing lower frequency wave phenomena.

### 2.3 Delay Network and Feedback Delay Network

Feedback Delay Networks were first pioneered by Jot and Chaigne in 1991 and remains a state-of-the-art reverberation method [7][19]. Feedback Delay Networks and other delay networks methods aim to simulate reverberation through networks of delay lines and digital filters. For the purpose of generating high quality artificial reverberation, a reverberator based on a Feedback Delay Network has the advantage that it can independently tune the energy storage, damping, and diffusion components related to the reverberation. However, Feedback Delay Networks and other delay networks are not physically based so they do not necessarily model the RIR accurately. Instead, they aim to capture the perceptual quality of the RIR [18].

FDN has two major components: a set of delay lines and a feedback matrix. For a given sound source the FDN generates the artificial reverberant sound by continually looping the signal through a set of delay lines (represented as a diagonal delay matrix), and then mixing the delayed signal with a feedback matrix

(potentially with some additional filters in between).

An example of a full FDN is displayed below in Fig. 2.2 to clarify the interaction between the input audio signal ( $u(n)$ ) and the components of a FDN. Besides the feedback matrix and delay lines (denoted  $\{q_{i,j}\}$  and  $z^{-M_i}$  in the figure), a FDN can also have a set of input gains ( $b_i$ 's), output gains ( $c_i$ 's), and feedback gains ( $g_i$ 's). Optionally, the output from each delay line may also be pass through a low pass filter (not pictured). The remainder of section 2.3 will describe the different components and typical design of the these components for a FDN as described in [7][18].

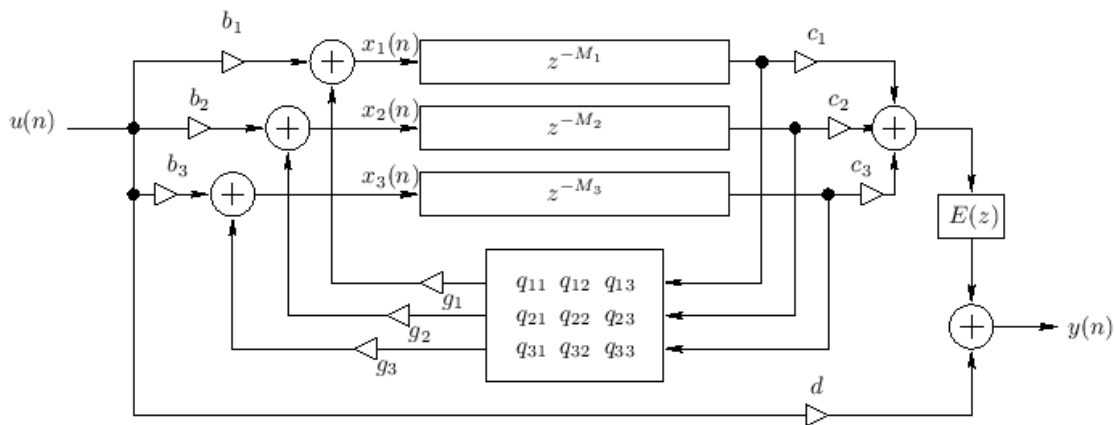


Figure 2.2: Example FDN structure from [7]

### 2.3.1 Feedback Matrix

In a FDN, the *feedback matrix* serves the role of mixing the delayed signals to simulate the aggregate effect of a sound signal's interaction with the surfaces of the enclosing environment. For this reason, the feedback matrix is sometimes also referred to as the scatter matrix. The Hadamard and Householder Feedback

Matrix are common choices for the feedback matrix in FDN and are well studied, but generally we can choose any unitary matrix to ensure stability for the FDN [18]. A Hadamard matrix is a square orthogonal matrix whose entries are either 1 or -1. Furthermore, it is known that for a Hadamard matrix of order  $n \geq 4$ ,  $n$  is divisible by 4. Although there are no known formula for constructing all possible Hadamard matrices, there are known methods for constructing Hadamard matrices of a specific structure [1]. An example of of a Hadamard matrix for  $n = 4$  would be:

$$H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}$$

The Householder matrix represents a reflection transformation about the hyperplane defined by the vector  $u$  [6] and is defined as

$$H_u = I - \left( \frac{2}{u^T u} \right) uu^T \quad (2.1)$$

### 2.3.2 Delay Line and Length

The delay line in the FDN specifies the amount of time that a signal that is passed through it gets “delayed,” and the number of delay line that the FDN has is often referred to as the number of tap the FDN has.

The delay line length should roughly be around the “mean free path” given approximately as  $\bar{d} = 4V/S$ , due to Sabine, where  $S$  is the total surface area of the room and  $V$  its enclosing volume [10] [20]. More precisely each delay line length,  $M_i$ ,

should be chosen to be mutually prime (to maximize the number of samples that the lossless reverberator prototype must be go through before the impulse response repeats) and chosen to ensure a sufficiently high mode density in all frequency bands (typically, we want  $M \geq 0.15t_{60}f_s$  where  $M = \sum_{i=1}^N M_i$ ).

To generate a set of prime power delay-line length, a common schema is to parameterize delay line length as an integer power of a distinct prime number; where the power is chosen as

$$m_i = \left\lfloor 0.5 + \frac{\log(M_i)}{\log(p_i)} \right\rfloor$$

for a given prime  $p_i$ , yielding the final set of delay line length as  $\{p_i^{m_i}\}$ .

In practice, to generate a good prime power delay line, we should ensure that the minimum delay line length roughly corresponds to the minimum acoustic ray length in the reverberator (that is, the desired delay time between the sound source and receiver for a given the sampling frequency). Similarly, we should bound the maximum delay line length to correspond to the maximum acoustic ray length (“room size”) [18].

## 2.4 Important Perceptual Metrics for Reverberation

To compare how similar two room impulse responses are to one another in terms of how an individual perceives them, it is useful to be able to quantify any perceptual differences. The following features are considered to be good metrics for measuring the perceptual accuracy of the generated impulse response for a given sampling frequency [8]:

- $t_{60}(f)$  - desired reverberation time at each frequency  $f$ , used as a measure of perceived reverberation time. It is defined as the time it takes for the sound level in the room to decrease by 60 dB [16]. The Energy Decay Curve (EDC) defined as

$$\text{EDC}(t) = \int_t^{\infty} h(\tau) d\tau \quad (2.2)$$

is often used to compute  $t_{60}$  since it decays more smoothly than the impulse response envelope [18].  $t_{60}$  may be approximated by Sabine's Formula [10] [20]:

$$T = \frac{0.049V}{S\bar{\alpha}} \quad (2.3)$$

for room dimensions measured in feet, where  $V$  is the volume of the room,  $S$  is the boundary surface area, and  $\bar{\alpha}$  is the average absorption coefficient given as

$$\bar{\alpha} = \frac{1}{S} \sum_{i=1}^n s_i \alpha_i \quad (2.4)$$

where  $s_i$  is the area of a boundary surface and  $\alpha_i$  is the absorption value of the corresponding boundary surface.

- $G^2(f)$  - signal power gain at each frequency
- $C_{50}(f)$  or  $C_{80}(f)$  - clarity or early-to-late index (related to the direct-to-reverberant-ratio), important for perception of room size and perception of sound quality [12]. It can be defined as

$$C_{80} = 10 \log_{10} \left[ \frac{\int_0^{80\text{ms}} h^2(t) dt}{\int_{80\text{ms}}^{\infty} h^2(t) dt} \right] \quad (2.5)$$

where  $h(t)$  is the impulse response.  $C_{50}$  is similarly defined [16].

Other features from the impulse response, such as the Mel-frequency cepstral coefficients (MFCC) and signal power envelope, can also serve as perceptual comparison metrics, though less common. Each of these metrics are computed for an individual impulse response and are then compared somehow base on their differences. Another way to compare the impulse response is with the spectral distortion and signal-to-distortion ratio. These two metrics can be use to directly compare two impulse responses by treating one of the impulse response (such as the FDN generated impulse response) as some distortion of the other impulse response.

## 2.5 Automatic Tuning of Feedback Delay Network

When Jot and Chaigne first propose FDN, the various components of FDN has to be manually designed and tuned for each room before a plausible late reverberation can be generated from the FDN. The FDN tuning is done in such a way to match some perceptual metrics (see Fig. 2.3 for example). Typically, FDN are designed for a given reverberation time at several frequencies.

The idea of automatically tuning a FDN was first proposed in [2] and then further refined in [3]. The core idea behind these automatic tuning methods is to tune the FDN to match a given RIR by using the Genetic Algorithm to optimize some loss function with respect to the FDN parameters describing the FDN. The loss function is a function of the given RIR and the generated RIR produced by the FDN, and should ideally model decrease in perceptual differences between the

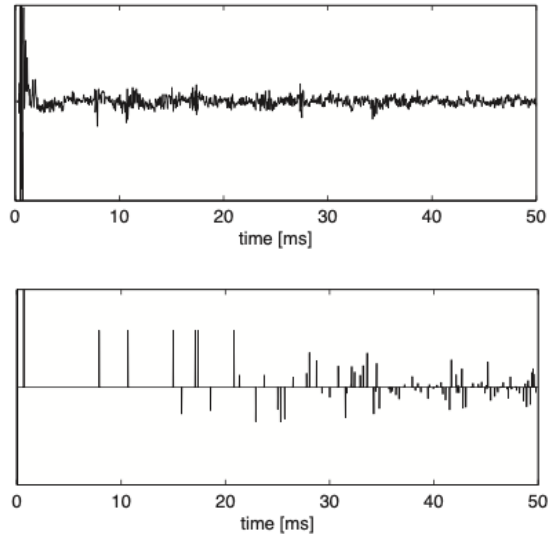


Figure 2.3: Example of a simulated impulse response from a FDN matched to have the same energy as the measured IR [11].

two RIR's as the loss approaches zero. To ensure that perceptual differences are captured in the loss function, perceptual metrics are used in formulating possible loss functions.

A subset of these perceptual metrics described in section 2.4 have been explored are compared in [3] for automatic tuning of FDN. The choice for the best set of perceptual metrics to use in the automatic tuning of FDN is explored to a limited extent in this thesis, though it is outside of the main scope of the thesis.

## Chapter 3: Methodology

### 3.1 Automatic FDN Construction from Room Parameters

We aim to construct a FDN in real-time that produces an impulse response that perceptually matches the impulse response of a given room described by a set of room parameters. A possible approach to accomplish this goal is to break the goal into two sub-problems. The first sub-problem involves computing the RIR of the parameterized room and computing some perceptual metric to represent the RIR. The second sub-problem then involves finding a FDN that produces an impulse response that matches the same perceptual metric. Here we propose an initial general method to solving the problem by combining the results from the two sub-problems to build a data set based on the correspondence between the set of room parameters and FDN parameters. The data set can then be used to train a regression model to infer the FDN parameters from a given set of room parameters. The general method is outlined below (Fig. 3.1):

1. Generate a set of RIR parameterized by a set of parameters describing a room
2. Apply optimization algorithm to generate a matching FDN for each RIR over a set of FDN parameters with a perceptual loss function



3. Train a regression model to learn mapping between room parameters and FDN parameters
4. Refine inferred FDN to generate late reverberation (optional)
5. Deploy regression model to infer a plausible FDN for each room of interest

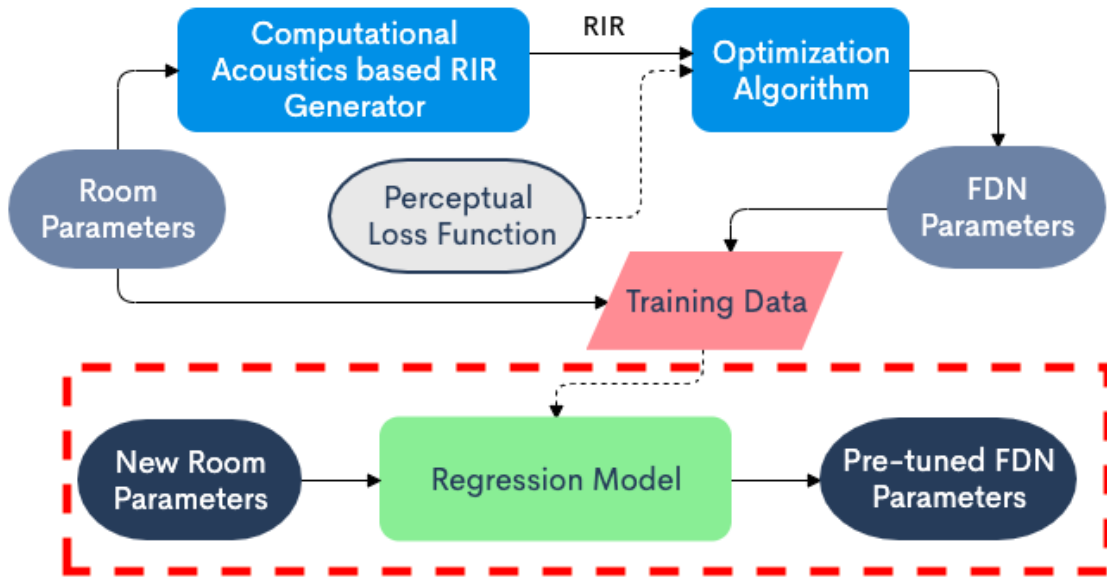


Figure 3.1: Flow chart of the proposed general method. The final regression model to deploy for inference is boxed in red.

One possible implementation approach for our proposed method using a shoe box room Binaural Room Impulse Response (BRIR) generator, Genetic Algorithm, and Support Vector Machine Regression is described in the rest of this section.

### 3.2 Obtaining the Room Impulse Response

BRIR simulation for a box room is used to generate a set of room impulse response corresponding to a room parameterized by its room dimensions (length, width, and height) and reflection coefficient of each of the six walls. Once the BRIR

is simulated, the BRIR corresponding to the “left ear” is chosen to be use as the RIR, and the reverberation time and other possible perceptual metrics can be computed from this RIR. The particular method used to generate the BRIR is described in [21]. Note that a box room is used here for ease of parameterization. This process can be generalize further by simulating BRIR for rooms with more complex geometries so long as there is a reasonable way to parameterize the complex geometries of the room.

### 3.3 Automatic Design of FDN

The automatic design of FDN can be frame as an inverse problem: given an impulse response for some room, recover a set of parameters that would describe a FDN that can produce the same matching impulse response (as close as possible).

Previous approach to solving this problem requires the use of Genetic Algorithm to search for the FDN parameters, given an objective function whose optima correspond to a match of some chosen perceptual metric. In that sense the inverse problem is solved through global optimization. Genetic Algorithm is a viable optimization method for this case because it does require the gradient of the perceptual loss function. Working with the gradient of the loss function is tricky because the loss function is a function of the impulse response generated from the FDN. This would require finding derivatives of a recurrent function which can be difficult in practice. Automatic differentiation or numerical differentiation might use apply to utilize gradient-based optimization methods and can be explore in later works.

We take a modified approach to tune the FDN based on work by [3]. As in their approach, the impulse response is generated at a sampling rate of 48kHz and the first 4096 sample are directly taken from the original matching impulse response. The main modification to the optimization process is that we use a four-tap FDN, the perceptual metric we use for the matching is different (see Section 3.4), and instead of constraining tap delay line length be to integer values that cover a scale of 1:2.5 and with the longest value corresponding to 100 milliseconds, we just have the constraint that the integers be mutually prime. This can be achieved by optimizing over a set of integers corresponding to the exponent component of the delay length and having a fixed set of prime basis:  $\{2, 3, 5, 7\}$ . Further, we do not modify the input and output gains, as well as the low-pass filter. In summary, we optimize over a parameterization of the FDN given by  $\{g, m_1, m_2, m_3, m_4\}$  where  $g$  is the feedback gain and  $m_i$ 's are the delay line length exponents. We also try out another parameterization where we only optimize based on the input gain ( $b$ ), output gain ( $c$ ), and feedback gain ( $g$ ) of the FDN.

### 3.4 Choice of Perceptual Metric

Some options explored by [3] so far for suitable perceptual metrics are ISO acoustic metrics (such as clarity index) matching, energy decay (EDC) matching, MFCC matching, and power envelope matching. For simplicity and as a proof-of-

concept, we chose the following loss function:

$$\mathcal{L}(\text{IR}, \text{IR}_{FDN}) = \frac{8}{10}(t_{60}(\text{IR}) - t_{60}(\text{IR}_{FDN}))^2 + \frac{2}{10}(C_{50}(\text{IR}) - C_{50}(\text{IR}_{FDN}))^2 \quad (3.1)$$

where IR is the impulse response we want to match,  $\text{IR}_{FDN}$  is the impulse response generated from the FDN, and  $t_{60}(\cdot)$  and  $C_{50}(\cdot)$  are the reverberation time and clarity index computed for each respective impulse response.

### 3.5 SVM Regression for Generating FDN Parameters

To achieve real time reverberation generation for a given room, just using the Genetic algorithm to construct the FDN parameters would not be sufficient because the optimization takes a non-trivial amount of time to compute. Thus, ideally we would want to be able to generate the FDN parameters in real time as soon as information about the room becomes available. To achieve this we can train a Support Vector Machine (SVM) regression model, and run inference on the trained model to predict the desired set of FDN parameters from the given room parameters in real time.

To train the SVM model would first require some training data. This training data can be generated first using the BRIR simulator and then with the Genetic algorithm applied to build correspondence between the room parameters that yield a particular RIR and a suitable FDN.

Two sets of training data were generated. The first data set consist of RIR generated from rooms with length and width varying between 3 to 8 meters and

height of 2.7 meters. The reflection coefficients for the walls are fixed with the side wall all set to the reflection coefficient and the top and bottom coefficient differs. Similarly, the second data set consist of RIR generated from rooms with length and width varying between 22 to 27 meters and height of 8 meters. The reflection coefficients for the walls are set the same way as the previous data set. We refer to the first data set as the small room data set and the second data set as the large room data set.

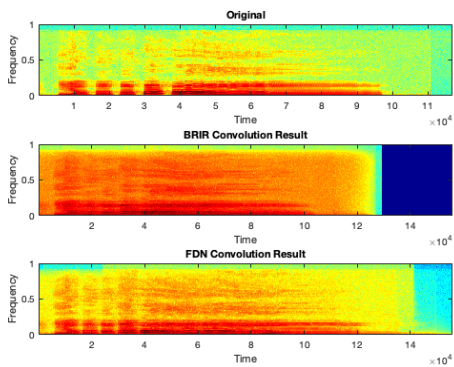
For each impulse response generated, we compute the reverberation time and clarity index and then use the automatic FDN tuning approach described above to generate the corresponding FDN parameters that “best” matches the impulse response in terms of the perceptual loss function [3.1](#). This establishes a correspondence between the room parameters that generated the impulse response with the FDN parameters. This correspondence can be use as training data for the SVM regression.

## Chapter 4: Results

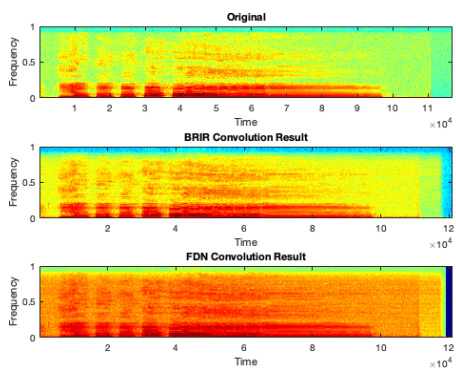
### 4.1 Genetic Algorithm Optimization Loss Value

Three sample results from the RIR matching using Genetic Algorithm is displayed below (Fig. 4.1(a), 4.1(b), 4.1(c)) in decreasing order of final loss value as calculated by Equation 3.1. The figures display the spectrogram of the reverberated sound produced using the simulated RIR compared to the spectrogram of the reverberated sound produced from the RIR generated from the matching FDN.

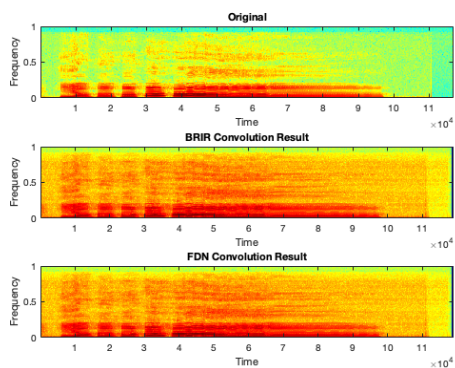
The visualization of the spectrogram and sound produced from the two RIR demonstrates visually the difference between the dry audio and the reverberated sound. It also illustrates visually how loss value effects the matching of two RIR.



(a) Loss value of 18.3822



(b) Loss value of 2.4727



(c) Loss value of  $1.0042 \times 10^{-4}$

Figure 4.1: Spectrogram of sound signal convolved with RIR and FDN produced RIR for various loss values. In each set of three spectrogram, the top is the original audio, middle is the result produced from the simulated RIR and bottom is the result produced from FDN generated RIR.

## 4.2 SVM Results

We want to build a regression model to map some parameterization of a boxed room to the corresponding parameterization of the FDN that would produce a perceptually identical room impulse response. In the impulse generation process, we have a simplified parameterization of the room given by  $(\ell, w, h)$  where  $\ell$ ,  $w$ , and  $h$  are the length, width, and height of the boxed room and the reflection coefficient of

each wall is described by a set of fixed  $\gamma$  taken to be in the range 0 to 1. To represent the FDN we parameterize it as  $(g, m_1, m_2, m_3, m_4)$  where  $g$  is the feedback gain that is then multiplied with the feedback matrix,  $\Lambda$ , and  $m_i$  is the exponent power we raise the prime basis to for our prime delay line. For the four tap FDN we are modeling, the prime bases used are 2, 3, 5, and 7. We alternatively parameterize the FDN as  $(g,b,c)$  where  $g$  is the feedback gain,  $b$  is the input gain, and  $c$  is the output gain.

With the genetic algorithm we were able to establish more than 100 correspondences so far between the set of room and FDN parameter for both the large room data set and the small room data set.

#### 4.2.1 Small Room Data Set Results

For the small room data set, we trained four sets of SVM regression model: for each parameterization of the FDN, we trained a SVM model with the Gaussian (RBF) kernel and a SVM model with the polynomial kernel. The SVM regression implementation from Matlab is used with the hyperparameters for the SVM model set to be determined automatically. For each model we report the mean squared error for each of the prediction of each FDN parameters. The mean squared error by itself is not very informative, rather the loss function described in Equation 3.1 evaluated over the predicted FDN is much more indicative of the perceptual relevance of the result. Thus, we also predict the perceptual loss value for each model.



The training and testing results for the first FDN parameterization ( $g, m_1, m_2, m_3, m_4$ ) is tabulated in Table 4.1 and 4.2 below. The training data is restricted to ones with loss value less than 0.8 from the initial Genetic Algorithm matching, and the remaining data serves as the testing data set. The threshold is set in such a way to ensure the training data set is approximately of the same size across all the models trained

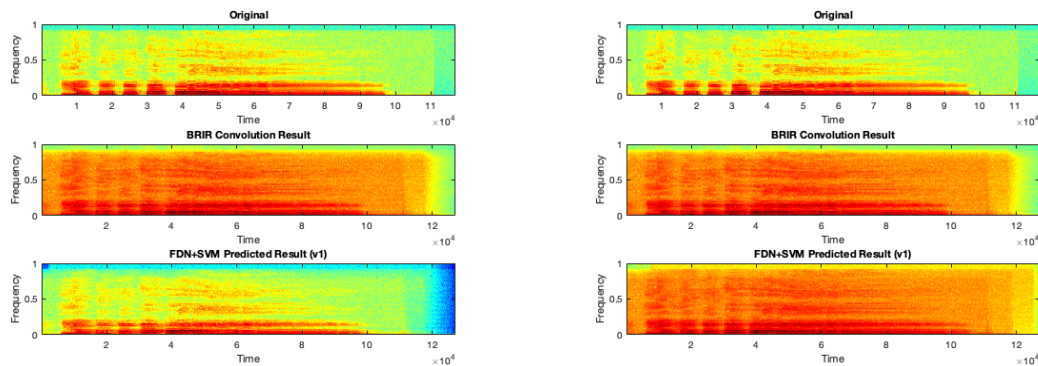
| Regression Model | FDN Parameters MSE |        |        |        |        | Perceptual Loss Stat. |        |
|------------------|--------------------|--------|--------|--------|--------|-----------------------|--------|
|                  | $g$                | $m_1$  | $m_2$  | $m_3$  | $m_4$  | Mean                  | Median |
| RBF Kernel SVM   | 0.0727             | 2.0345 | 4.8966 | 1.1724 | 1.0690 | 1.9955                | 2.1470 |
| Poly. Kernel SVM | 0.0284             | 4.1379 | 4.9310 | 0.7586 | 0.4828 | 45.7648               | 2.3623 |

Table 4.1: Training performance for SVM regression model for FDN parameterization ( $g, m_1, m_2, m_3, m_4$ ) for the small room data set.

| Regression Model | FDN parameters MSE |         |         |        |        | Perceptual Loss Stat. |         |
|------------------|--------------------|---------|---------|--------|--------|-----------------------|---------|
|                  | $g$                | $m_1$   | $m_2$   | $m_3$  | $m_4$  | Mean                  | Median  |
| RBF Kernel SVM   | 0.0840             | 16.2254 | 12.0704 | 3.2113 | 1.4225 | 1.5700                | 1.0308  |
| Poly. Kernel SVM | 0.0861             | 22.7042 | 13.8592 | 4.5915 | 1.3099 | 246.0664              | 20.8343 |

Table 4.2: Testing performance for SVM regression model for FDN parameterization ( $g, m_1, m_2, m_3, m_4$ ) for the small room data set.

Sample spectrogram of the audio output from the FDN predicted from the SVM model is displayed in Fig. 4.2. The spectrogram of the dry audio and the reverberated audio obtained from convolution with RIR is also included for reference. Note that the sample output is not necessarily representative of how well the model perform.



(a) Sample result from SVM with Gaussian Kernel      (b) Sample result from SVM with Polynomial Kernel

Figure 4.2: Sample output from the SVM regression model trained on the small room data set with FDN parameterized as  $(g, m_1, m_2, m_3, m_4)$ . In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom).

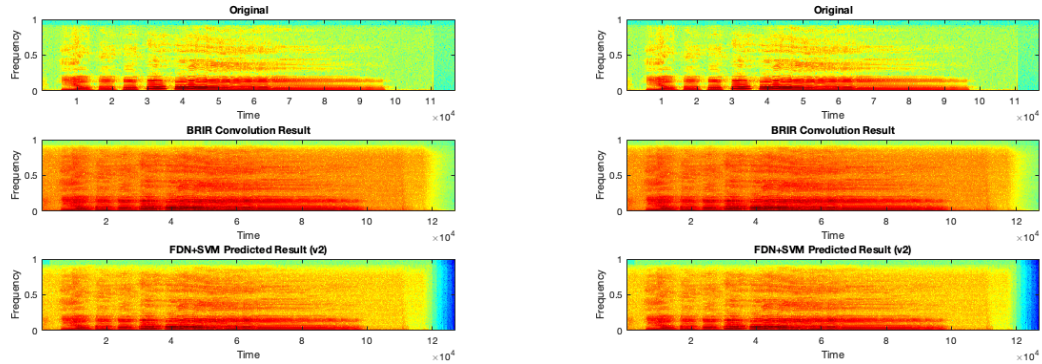
Now we shift to reporting the SVM regression result of predicting the FDN parameterized by  $(g,b,c)$ . As before, the training and testing results for the second FDN parameterization is tabulated in Table 4.3 and 4.4 below. The training data is restricted to ones with loss value less than 0.2 from the initial Genetic Algorithm matching, and the remaining data serves as the testing data set. Sample spectrogram of the audio output from the FDN predicted from the SVM model is displayed in Fig. 4.3 like before.

| Regression Model | FDN Parameters MSE |        |        | Perceptual Loss Stat. |        |        |        |
|------------------|--------------------|--------|--------|-----------------------|--------|--------|--------|
|                  | $g$                | $b$    | $c$    | Mean                  | Median | Min    | Max    |
| RBF Kernel SVM   | 0.0083             | 0.0040 | 0.0047 | 0.0676                | 0.0343 | 0.0010 | 0.5456 |
| Poly. Kernel SVM | 0.0101             | 0.0053 | 0.0053 | 0.0713                | 0.0378 | 0.0003 | 0.3636 |

Table 4.3: Training performance for SVM regression model for FDN parameterization  $(g,b,c)$  for the small room data set.

| Regression Model | FDN Parameters MSE |        |        | Perceptual Loss Stat. |        |        |        |
|------------------|--------------------|--------|--------|-----------------------|--------|--------|--------|
|                  | $g$                | $b$    | $c$    | Mean                  | Median | Min    | Max    |
| RBF Kernel SVM   | 0.0734             | 0.0265 | 0.0051 | 0.1519                | 0.0622 | 0.0018 | 0.8613 |
| Poly. Kernel SVM | 0.0761             | 0.0320 | 0.0067 | 0.1332                | 0.0721 | 0.0016 | 0.7356 |

Table 4.4: Testing performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the small room data set.



(a) Sample result from SVM with Gaussian Kernel

(b) Sample result from SVM with Polynomial Kernel

Figure 4.3: Sample output from the SVM regression model trained on the small room data set with FDN parameterized as ( $g, b, c$ ). In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom).

## 4.2.2 Large Room Data Set Results

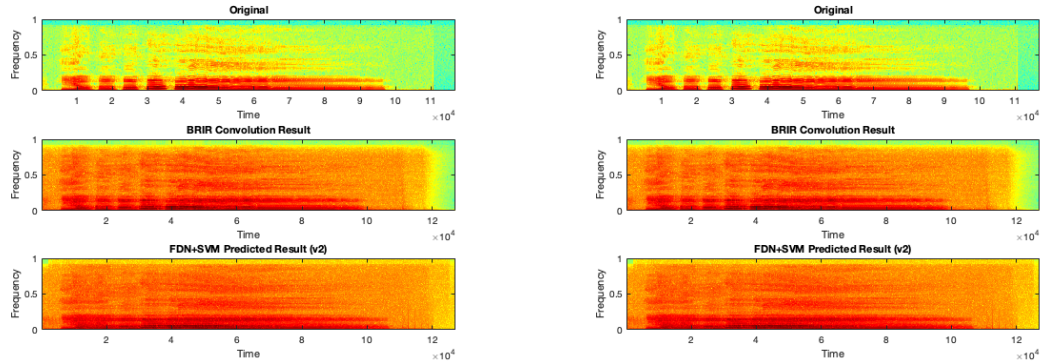
For the SVM regression model trained on the large room data set, we also report its mean squared training error and loss function statistics in Table 4.5 and 4.6. The threshold for the training set is now set to ones with loss value less than 0.7 from the initial Genetic Algorithm matching, and the remaining data serves as the testing data set as before. Sample spectrogram of the audio output from the FDN predicted from the SVM model is displayed in Fig. 4.4. Models were only trained for the second FDN parameterization for the large room data set.

| Regression Model | FDN Parameters MSE |        |        | Perceptual Loss Stat. |         |         |         |
|------------------|--------------------|--------|--------|-----------------------|---------|---------|---------|
|                  | $g$                | $b$    | $c$    | Mean                  | Median  | Min     | Max     |
| RBF Kernel SVM   | 0.0045             | 0.0172 | 0.0113 | 31.2218               | 31.6700 | 25.0280 | 34.5177 |
| Poly. Kernel SVM | 0.0062             | 0.0159 | 0.0110 | 31.1467               | 32.0197 | 21.9203 | 47.1815 |

Table 4.5: Training performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the large room data set.

| Regression Model | FDN Parameters MSE |        |        | Perceptual Loss Stat. |         |         |         |
|------------------|--------------------|--------|--------|-----------------------|---------|---------|---------|
|                  | $g$                | $b$    | $c$    | Mean                  | Median  | Min     | Max     |
| RBF Kernel SVM   | 0.0075             | 0.0647 | 0.0397 | 31.4420               | 31.7469 | 26.3002 | 34.7566 |
| Poly. Kernel SVM | 0.0082             | 0.0569 | 0.0360 | 32.1311               | 31.8487 | 22.4158 | 62.6516 |

Table 4.6: Testing performance for SVM regression model for FDN parameterization ( $g, b, c$ ) for the large room data set.



(a) Sample result from SVM with Gaussian Kernel

(b) Sample result from SVM with Polynomial Kernel

Figure 4.4: Sample output from the SVM regression model trained on the large room data set with FDN parameterized as ( $g, b, c$ ). In each set of three spectrogram, the original audio (top), result produced from the simulated RIR (middle) and result produced from the FDN predicted by the SVM (bottom).

### 4.3 Real-time System Demonstration

To demonstrate the feasibility of approximating the late-reverberation in real-time using our proposed method, a mock demo was implemented in C/C++ demonstrate the audio stream processing capability of FDN. The demo consists of two system processes. The first process sends sample audio data through a pipe to sim-

ulate a dummy audio stream. The second process reads from the pipe and process the audio stream in batches of 4800 samples and outputs the resulting audio that has been passed through the FDN.

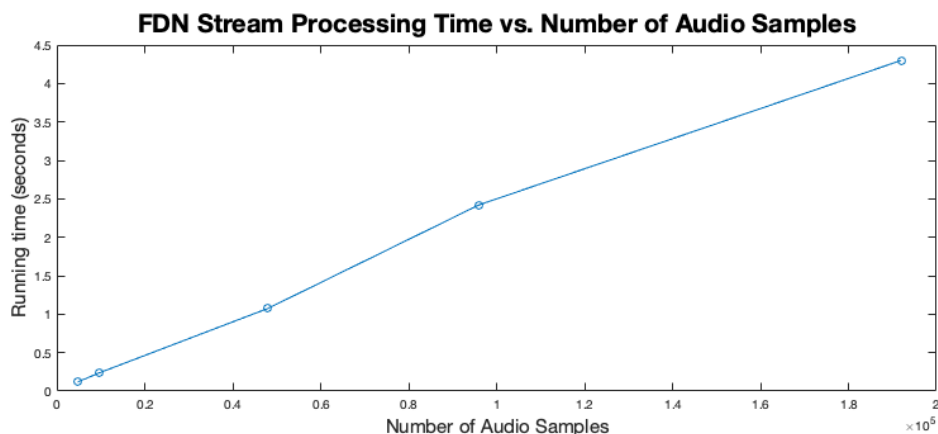


Figure 4.5: Plot of the total amount of time the mock demo took to process different number of audio samples.

The running time of the mock demo run on a 1.6 GHz Dual-Core Intel Core i5 processor is displayed in Fig. 4.5. This suggests that FDN should be able to stream and process audio in real-time, depending on the sampling frequency of the audio, tap-delay line lengths, and the processor speed.

At the time of this writing, a full virtual demo of our pre-tuned FDN is also under development using [14], which is a real-time virtual environment rendering system originally developed in the Spatial Auditory Displays Lab at NASA Ames Research Center.

## Chapter 5: Discussions

### 5.1 Quality of the Training Data Set

Our method of obtaining FDN parameters that perceptually matches some target impulse response is to set an objective function to compare the FDN generated impulse response with the target and then optimize for that objective with respect to the FDN parameters. This matching process is done for each RIR we generated from a set of room parameters, thus the quality of our training data set for the SVM regression model is limited by how closely the Genetic Algorithm was able to tune the FDN parameters.

In practice, we see from the results in section [4.1](#) that the matching computed from the Genetic Algorithm is not necessarily always good since there are already some visually distinct differences between the FDN produced RIR and the simulated RIR when the loss value is around 2. As with any optimization algorithm, Genetic Algorithm can occasionally run into issues with finding sub-optimal solution when trying to minimize the loss function. This can occur if the Genetic Algorithm did not converge within the given maximum iterations (or “generations” in the specific context of Genetic Algorithm) or if the optimization got stuck in local-minima. In the context we are working with, a sub-optimal result means that the set of FDN

parameters obtained from the Genetic Algorithm corresponds to objective function that is sufficient large enough to yield a noticeable perceptual difference between the generated impulse response and the desired impulse response.

There are practical ways to improve the sub-optimal results, however, under time and resource constraint we would have to settle and deal with these sub-optimal results. Pre-mature convergence can be overcome with high probability by applying the Genetic algorithm multiple times and taking the best result out of all the run. We have done this a couple time to construct our data set, but some sub-optimal results still remain. Meanwhile, having results that have not yet converge before the maximum number of generations can be overcome simply by increasing the maximum allowed generations or setting the algorithm to terminate only when certain objective threshold is met. This, however, can significantly increase the amount of time it takes to optimize for one impulse response, and the amount of increase may or may not be too much to handle. We have tried optimizing up to 15 generations with a population size of 20. We have applied these methods to improve our data set quality and have driven down the loss function value calculated from each optimization to be less than one. User listening tests may be necessary to confirm whether the FDN produced from the SVM trained on the data set is sufficiently good for application purposes.

## 5.2 SVM Regression Performance

In section 4.2, the mean squared training error serves mainly as an indication of the SVM model’s ability to learn the patterns captured in the training data. We see evidence of this through the SVM model for the large room data set where despite low mean squared training error the loss value remains high. While this training error is not necessarily indicative of the perceptual relevance of the SVM model, it can be useful for comparing between regression models.

The loss function value of the RIR generated from the predicted FDN is more directly relevant to the perceptual performance of the SVM regression in actual audio rendering application. This can be seen in section 4.1 where we see that the spectrogram of the reverberant sound produced from the optimized FDN more visually resemble the spectrogram of the reverberant sound produced from the simulated RIR with lower loss values. Base on the loss value the polynomial kernel SVM model trained on the small room data set has a higher variance in terms of being able to predict a good set of FDN parameters for the given room dimension compared to the Gaussian kernel SVM model.

When predicting the FDN with the first parameterization, notice that the mean squared training error of the  $m_i$ ’s are significantly larger than that of the feedback gain parameter  $g$ . This suggests that the regression model is not learning the  $m_i$  exponents well. The  $m_i$ ’s parameters seems more difficult to learn likely because the set of integer powers we optimize over is very small and contain patterns that is hard to disambiguate from overlap. It is also possible that there is little



correlation between the individual value of the  $m_i$  values with the perceptual metrics we are matching, and that it is the aggregate behavior of the  $m_i$  exponents that has relation to the perceptual metrics.

Another interesting result from the SVM regression worth pointing out is the high loss value for the model trained and evaluated over the large room data set. This is in spite of a mean training error lower than the mean training error for the SVM model trained on the small room data for predicting  $(g, m_1, m_2, m_3, m_4)$ . A possible explanation for this poor result is that a four tap FDN is not sufficiently complex enough to model the reverberation of a large room; perhaps a eight tap FDN would be more sufficient for modeling larger rooms. Another possibility is that the delay line lengths chosen is not appropriate for modeling the larger rooms.

Out of the different parameterization and data set we considered, training the SVM model on the small room data set to predict the feedback, input, and output gains parameters demonstrate that it is possible to construct a model that can predict good pre-tuned FDN that closely model the impulse response of a given room.

### 5.3 Computational Cost Considerations

As seen in Section 4.3, the FDN is able to process the audio stream in approximately linear time with respect to the duration of the stream. This means depending on the quality of the audio we want to process, that is, its sampling rate, we can potentially process each signal before the next signal arrives: yielding

real-time performance.

What this also mean is that the time it will take to run Genetic Algorithm is also approximately linear with respect to the length of the audio signal (or more specifically, the length of the impulse response). However, the constant factor can be quite large and have a significant impact on the running time. The constant factor associated with this linear running time has to do with the maximum length of our delay line and also the population size and maximum number of generations allowed for the Genetic Algorithm implementation. This in practice mean that the Genetic Algorithm can take time on the order of half an hour to run for a population of 10 and max generation of 5. This make the optimization procedure using the Genetic Algorithm relatively expensive, especially when scaled to a large data set size or considering that the algorithm is not guaranteed to converge to a good result within the maximum set generation.

## Chapter 6: Conclusion

### 6.1 Summary

Through the course of this thesis work we have examine previous works done to generate artificial reverberation. On the one hand, we have convolutional and computational acoustic approaches that are capable of producing highly accurate reverberant sound, but at a high computational cost not practical for real-time applications. On the other hand, we have delay network based approach that is computationally efficient, but produce a lower quality reverberant sound. With the goal of producing higher quality reverberant sound in real-time, we developed a data-driven framework for enabling real-time approximation of late-reverberation. This approximation method can be combined with existing efficient methods for rendering the direct path and early reflections of a sound source to render the full reverberant audio source in real-time, as shown in the mock demo. This data-driven approach can be view as a hybrid method that takes advantage of the high quality RIR generated offline using computational acoustics method to quickly infer a plausible FDN for efficient rendering. This is useful for enhancing the realism of augmented and virtual reality applications where audio signal needs to be streamed to the user in real-time, such as virtual video conferencing and concert performance

broadcasting. To conclude, we make several suggestions on possible future direction related to the thesis.

## 6.2 Possible Future Work

One area with rooms for improvement is to improve the speed and effectiveness of the optimization process used to build the training data set. Because some of the room dimension input are similar, an optimal output for one set of impulse response matching can be a good candidate initial candidate for another that is similar. This mean we can potentially achieve speed up for the optimization process from joint optimization of multiple impulse response using the Genetic Algorithm, allowing us to more efficiently build up a larger and more representative training data set and improve the data set quality. The choice of using the Genetic Algorithm itself is also open for exploration, it is possible that gradient based optimization methods will be more suitable for the optimization process involved in matching a FDN to a given RIR.

Similarly, the choice of regression model for predicting the FDN parameters from the room parameters can be explored as well. Once a sufficiently large training set is constructed, it might even be possible to apply neural networks for handling the regression task.

Finally, it is known that with a given HRTF, it is possible to combine two FDN to simulate the Binaural Room Impulse Response (BRIR). Rendering a sound source using a person's HRTF and BRIR provides additional value over using just

the RIR because it enables a higher level of personalization and realism for the listener. More sophisticated FDN can be considered to improve the realism of the reverberant audio even further, so long as the more complex FDN model still remain efficient for evaluation.

## Bibliography

- [1] Richard A Brualdi, Shmuel Friedland, and Victor Klee. *Combinatorial and graph-theoretical problems in linear algebra*. Vol. 50. Springer Science & Business Media, 2012.
- [2] Michael Chemistruck, Kyle Marcolini, and Will Pirkle. “Generating matrix coefficients for feedback delay networks using genetic algorithm”. In: *Audio Engineering Society Convention 133*. Audio Engineering Society. 2012.
- [3] Jay Coggin and Will Pirkle. “Automatic Design of Feedback Delay Network Reverb Parameters for Impulse Response Matching”. In: *Audio Engineering Society Convention 141*. Audio Engineering Society. 2016.
- [4] Nail A Gumerov and Ramani Duraiswami. *Fast multipole methods for the Helmholtz equation in three dimensions*. Elsevier, 2005.
- [5] Brian Hamilton and Stefan Bilbao. “FDTD methods for 3-D room acoustics simulation with high-order accuracy in space and time”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 25.11 (2017), pp. 2112–2124.
- [6] Alston S Householder. “Unitary triangularization of a nonsymmetric matrix”. In: *Journal of the ACM (JACM)* 5.4 (1958), pp. 339–342.
- [7] Jean-Marc Jot and Antoine Chaigne. “Digital delay networks for designing artificial reverberators”. In: *Audio Engineering Society Convention 90*. Audio Engineering Society. 1991.
- [8] J-P Jullien et al. “Spatializer: a perceptual approach”. In: *PREPRINTS-AUDIO ENGINEERING SOCIETY* (1993).
- [9] Stephen Kirkup. “The boundary element method in acoustics: a survey”. In: *Applied Sciences* 9.8 (2019), p. 1642.
- [10] Heinrich Kuttruff. *Room acoustics*. Crc Press, 2016.
- [11] Fritz Menzer. “Binaural reverberation using two parallel feedback delay networks”. In: *Audio Engineering Society Conference: 40th International Conference: Spatial Audio: Sense the Sound of Space*. Audio Engineering Society. 2010.
- [12] Andrzej Miśkiewicz et al. “Concert hall sound clarity: A comparison of auditory judgments and objective measures”. In: *Archives of Acoustics* 37.1 (2012), pp. 41–46.

- [13] James A Moorer. “About this reverberation business”. In: *Computer music journal* (1979), pp. 13–28.
- [14] NASA Spatial Auditory Displays Lab. *slab3d User Manual*. Version 6.8.3. May 1, 2020. URL: <http://slab3d.sonisphere.com>.
- [15] Takeshi Okuzono et al. “An explicit time-domain finite element method for room acoustics simulations: Comparison of the performance with implicit methods”. In: *Applied Acoustics* 104 (2016), pp. 76–84.
- [16] Thomas Rossing. *Springer handbook of acoustics*. Springer Science & Business Media, 2007.
- [17] Lauri Savioja and U Peter Svensson. “Overview of geometrical room acoustic modeling techniques”. In: *The Journal of the Acoustical Society of America* 138.2 (2015), pp. 708–730.
- [18] Julius O. Smith. *Physical Audio Signal Processing*. online book, 2010 edition. <http://ccrma.stanford.edu/~jos/pasp/>, accessed April, 2020.
- [19] Vesa Valimaki et al. “Fifty years of artificial reverberation”. In: *IEEE Transactions on Audio, Speech, and Language Processing* 20.5 (2012), pp. 1421–1448.
- [20] Robert W Young. “Sabine reverberation equation and sound power calculations”. In: *The Journal of the Acoustical Society of America* 31.7 (1959), pp. 912–921.
- [21] Dmitry N Zotkin, Ramani Duraiswami, and Larry S Davis. “Rendering localized spatial audio in a virtual auditory space”. In: *IEEE Transactions on multimedia* 6.4 (2004), pp. 553–564.