

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**CONDITIONAL GENERATIVE ADVERSARIAL NETWORK
BASED ROOM IMPULSE RESPONSE GENERATOR**



MEHMET PEKMEZCİ

A THESIS OF DOCTORATE
DEPARTMENT OF COMPUTER ENGINEERING

ADVISOR: ASSOC. PROF. DR. YAKUP GENÇ

JULY 2025

T.R.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL

**CONDITIONAL GENERATIVE ADVERSARIAL
NETWORK BASED ROOM IMPULSE RESPONSE
GENERATOR**

MEHMET PEKMEZCİ

A THESIS OF DOCTORATE
DEPARTMENT OF COMPUTER ENGINEERING

ADVISOR: ASSOC. PROF. DR. YAKUP GENÇ

JULY 2025

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

KOŞULLU ÇEKİŞMELİ AĞ TABANLI ODA YANKI İZİ
ÜRETECİ

MEHMET PEKMEZCİ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMAN: DOÇ. DR. YAKUP GENÇ

TEMMUZ 2025

DOCTORATE JURY APPROVAL FORM

A thesis submitted by Mehmet, Pekmezci, defended on 10/07/2025 before the jury formed with the 23/12/2024 date and 2024/67 numbered decision of the GTU Graduate Administration Board, has been accepted as a DOCTOR of PHILOSOPHY thesis in the Department of Computer Engineering.

JURY

MEMBER

(THESIS ADVISOR) : Assoc. Prof. Dr. Yakup GENÇ

MEMBER

: Prof. Dr. Mürvet KIRCI

MEMBER

: Prof. Dr. Fatih Erdoğan SEVİLGİN

MEMBER

: Prof. Dr. Mehmet GÖKTÜRK

MEMBER

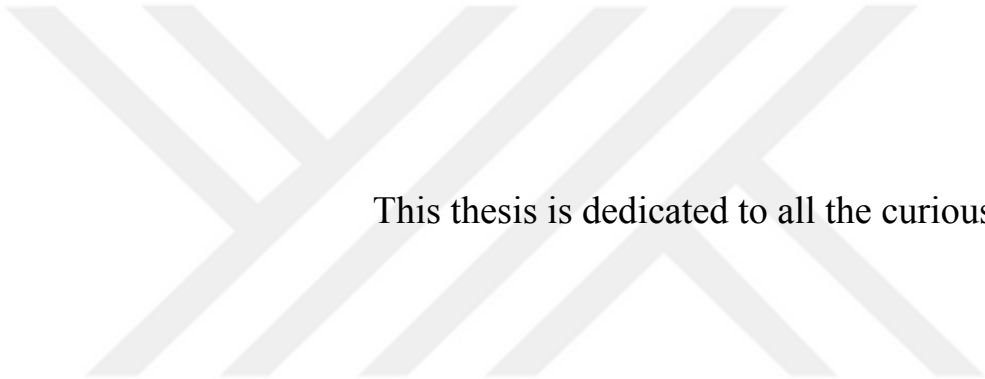
: Assist. Prof. Dr. Ali Köksal HOCAOĞLU

APPROVAL

Gebze Technical University Graduate Administration Board

...../...../..... date and/..... numbered decision.

SIGNATURE/SEAL



This thesis is dedicated to all the curious minds.

ABSTRACT

Far-field automatic speech recognition applications require addressing reverberation effects, either by dereverberating the received audio signals or by training the recognizer neural networks on reverberated sound data tailored to specific environmental conditions. Similarly, virtual and augmented reality applications demand accurate auralization of sound signals within virtual environments, adapting to specific sound source and receiver locations to create a natural auditory experience.

Additionally, applications such as room size and shape estimation, direction of arrival estimation, material detection within a room, listener envelopment evaluation in concert halls, inferring the location of reflecting surfaces, ultrasound tomography, water pipeline defect detection in buildings, and more require the ability to generate reverberated sound signals.

Reverberated sound signals can be generated using either sinusoidal functions or room impulse response (RIR) functions. While the sinusoidal function-based method often produces a mechanical-sounding reverberation, the RIR approach results in a much more natural-sounding reverberation.

Although RIRs can be directly measured in real-world environments, this is not always feasible—particularly in virtual reality settings or situations involving complex or inaccessible environments. In such cases, synthetic generation of RIRs becomes essential. To achieve this, we must account for the environment’s geometry (provided as a mesh), the positions of the loudspeaker and microphone, and the reflection coefficients of the materials present in the space. This approach enables the realistic simulation of reverberation in a variety of scenarios where direct measurement is impractical.

Room impulse response (RIR) generation techniques can be categorized into wave-based, ray-based, and statistical methods. Among these, deep neural network (DNN)-based methods fall under the statistical category. Compared to wave-based and ray-based approaches, DNN-based methods offer the best performance-to-complexity ratio, making them particularly suitable for applications requiring both efficiency and accuracy.

Generative Adversarial Networks (GANs), widely recognized for their success in image and sound generation tasks, are highly effective for generating RIRs. By leveraging GANs, we can synthesize realistic RIRs while reducing computational overhead, thereby addressing the limitations of traditional wave- or ray-based techniques. This makes GANs a compelling choice for tasks requiring high-quality RIR generation in complex or dynamic environments.

Previous works employing Generative Adversarial Networks (GANs) for generating room impulse responses (RIRs) have shown promise but also revealed two significant issues:

- Glitches in generated RIRs: Existing models introduce glitches in to the generated RIRs. These glitches can introduce distortions when the RIRs are convolved with the source sound, adversely affecting the resulting audio quality. In this study, we propose a method to mitigate these glitches, ensuring smoother and more natural-sounding RIRs.
- Coupled geometry and acoustic training: Existing models tightly integrate geometry embedding and acoustic modeling, requiring joint training of both components. This dependency limits the ability to generalize and generate RIRs for unseen types of geometries. In this work, we present a method to decouple the geometric and acoustic components of the neural network model, enabling independent training and improving flexibility for handling diverse room geometries.

By addressing these challenges, our approach not only enhances the quality of the generated RIRs but also increases the adaptability of the model to novel geometric configurations.

Keywords: Structural Similarity, Generative Adversarial Networks, Room Impulse Response, Graph Convolutional Networks, Mesh Transformer Autoencoder

ÖZET

Uzak alan otomatik konuşma tanıma uygulamaları, yankı etkilerini ele almayı gerektirir; bu ya alınan ses sinyallerini yankıdan arındırarak ya da belirli çevresel koşullara uygun şekilde yankılanmış ses verileri üzerinde tanıyıcı sinir ağlarını eğiterek gerçekleştirilir. Benzer şekilde, sanal ve artırılmış gerçeklik uygulamaları, sanal ortamlardaki ses sinyallerinin doğru şekilde işitselleştirmesini gerektirir. Bu, ses kaynağının ve alıcının konumlarına uyum sağlayarak doğal bir işitsel deneyim oluşturmaya hedefler.

Ayrıca, oda boyutu ve şekil tahmini, geliş yönü tahmini, bir odadaki malzeme tespiti, konser salonlarındaki dinleyici çevreleme değerlendirmesi, yansıtıcı yüzeylerin konumlarının çıkarılması, ultrason tomografisi, binalardaki su borusu arıza tespiti ve daha fazlası gibi uygulamalar, yankılanmış ses sinyallerinin üretilmesini gerektirir.

Yankılanmış ses sinyalleri, ya sinüzoidal fonksiyonlar ya da oda yankı izi (RIR) fonksiyonları kullanılarak üretilebilir. Sinüzoidal fonksiyon tabanlı yöntem genellikle mekanik bir yankı sesi üretirken, oda yankı izi yaklaşımı çok daha doğal bir yankı sesi sunar.

Oda Yankı İzi (Room Impulse Response, RIR), bir ortamda bulunan ses kaynağından çıkan sesin, aynı ortamda bulunan bir mikrofona gelinceye kadar, oda içindeki fiziksel ve geometrik koşullara göre değişimini tanımlamak için kullanılan “matematiksel fonksiyon parametresidir”. Burada kullanılan matematiksel fonksiyon genellikle Denklem 0.1’de görüldüğü gibi bir evrişimdir (konvolüsyondur). Evrişimin sol taraftaki parametresi, ses kaynağından çıkan ses, sağ taraftaki parametresi de oada yankı izidir, sonuç ise mikrofondan algılanan sestir.

$$(Ses\ kaynagindan\ cikan\ ses) * (RIR) = (Mikrofondan\ algilanan\ ses) \quad (0.1)$$

Aslında “Oda Yankı İzi” de bir ses dalgasıdır. Dolayısıyla bilgisayarda wav dosyası formatında saklanmaktadır. Yankı izini en iyi şekilde bir “ses darbesi” (impulse) kullanarak elde edebiliriz. Ses kaynağından ses darbesi şeklinde çıkan ses, ortamdaki diğer düşük seviyeli gürültü kaynaklarından kolaylıkla ayrışır.

Oda yankı izleri gerçek dünya ortamlarında doğrudan ölçülebilir olsa da, bu her zaman mümkün değildir—özellikle sanal gerçeklik ortamlarında veya karmaşık ya da erişilmesi zor ortamlarda. Bu tür durumlarda, oda yankı izlerinin sentetik olarak üretilmesi zorunlu hale gelir. Bunu başarmak için ortamın geometrisi (bir ağ olarak sağlanır), hoparlör ve mikrofona konumları ve ortamda bulunan malzemelerin yansıma katsayıları dikkate alınmalıdır. Bu yaklaşım, doğrudan ölçümün pratik olmadığı çeşitli senaryolarda yankı simülasyonunun gerçekçi bir şekilde gerçekleştirilmesini sağlar.

Oda yankı izi (RIR) üretim teknikleri dalga tabanlı, ışın tabanlı ve istatistiksel yöntemler olarak sınıflandırılabilir. Bunlar arasında derin sinir ağı (DNN) tabanlı yöntemler istatistiksel kategoriye girer. Dalga tabanlı ve ışın tabanlı yaklaşımlarla karşılaştırıldığında, DNN tabanlı yöntemler en iyi performans-kompleksite oranını sunar, bu da onları hem verimlilik hem de doğruluk gerektiren uygulamalar için özellikle uygun kılar.

Çekişmeli Üretken Ağlar (GAN’lar), görüntü ve ses üretimindeki başarılarıyla tanınmakta olup, oda yankı izi üretiminde de son derece etkilidir. Çekişmeli Üretken Ağlar sayesinde gerçekçi oda yankı izleri sentezlenirken hesaplama yükü azaltılabilir. Bu, geleneksel dalga veya ışın tabanlı tekniklerin sınırlamalarını aşar ve Çekişmeli Üretken Ağları, karmaşık veya dinamik ortamlarda yüksek kaliteli oda yankı izi üretimi gerektiren görevler için cazip bir seçenek haline getirir.

Çekişmeli Üretken Ağlar (GAN’lar) kullanılarak oda yankı izleri (RIR’ler) üretmeye yönelik önceki çalışmalar, umut verici sonuçlar göstermiş ancak aynı zamanda iki önemli sorunu ortaya çıkarmıştır:

- Üretilen Oda Yankı İzlerindeki Kusurlar: Bu kusurlar, oda yankı izlerinin kaynak ses ile evrişime tabi tutulduğunda bozulmalara neden olabilir ve sonuçtaki ses kalitesini olumsuz etkileyebilir. Bu çalışmada, bu kusurları azaltmak ve daha

pürüzsüz ve doğal sesli oda yankı izleri sağlamak için bir yöntem öneriyoruz.

- Geometri ve Akustik Eğitimde Bağlılık: Mevcut modeller, geometri gömülü kodlaması ve akustik modellemeyi sıkı bir şekilde entegre eder ve her iki bileşenin birlikte eğitilmesini gerektirir. Bu bağımlılık, yeni geometri türleri için oda yankı izlerinin geliştirilmesini ve üretilmesini sınırlar. Bu çalışmada, sinir ağı modelinin geometrik ve akustik bileşenlerini ayıran bir yöntem sunarak bağımsız eğitim yapılmasını sağlıyor ve farklı oda geometrileriyle başa çıkma esnekliğini artırıyoruz.

Bu zorlukların ele alınmasıyla, önerilen yaklaşım sadece üretilen oda yankı izlerinin kalitesini artırmakla kalmaz, aynı zamanda modeli yeni geometrik yapılandırmalara uyarlama yeteneğini de geliştirir.



Anahtar Kelimeler: Yapısal Benzerlik, Çekişmeli Üretken Ağlar, Oda Yankı İzi, Çizge Evrişimsel Ağları, Örgü Oto-dönüştürücü

ACKNOWLEDGEMENTS

I would like to thank the following people, without whom I would not have been able to complete this research, and without whom I would not have made it through my phd degree! The Gebze Technical University Computer Engineering Department academic staff members, especially to my supervisor Assoc. Prof. Dr. Yakup GENÇ, and steering committee members Prof. Dr. Fatih Erdoğan SEVİLGİN and Prof. Dr. Mürvet KIRCI, whose insight and knowledge into the subject matter steered me through this research. My colleagues Dr. Kürşat İNCE and Apdül YALIKAYA, who have supported me every time I need help. My company Havelsan A.Ş. for some of the equipments. And my biggest thanks to my family for all the support you have shown me through this research.



TABLE OF CONTENTS

	Page
ABSTRACT	vi
ÖZET	vii
ACKNOWLEDGEMENT	ix
TABLE of CONTENTS	x
LIST of ABBREVIATIONS and ACRONYMS	xii
LIST of FIGURES	xiii
LIST of TABLES	xv
1. INTRODUCTION	1
1.1. Motivations	1
1.2. Problem Definition	5
1.3. Contributions	7
1.4. Outline	8
2. LITERATURE REVIEW	9
2.1. Room Impulse Response Measurement	12
2.2. Room Impulse Response Generation	15
2.2.1. Ray Based Methods	17
2.2.2. Wave Based Methods	18
2.2.3. Neural Network-Based Mehods	22
2.2.4. Evaluation of SSIM Loss Function	28
2.2.4.1. Structural Similarity Index Measure (SSIM)	28
2.2.4.2. Audio Similarity Measurement using SSIM	34
2.2.4.3. Strengths and Weaknesses of SSIM in Audio Data	35
2.2.5. Seperating Geometric and Acoustic Models	39
2.3. Summary	46
3. GTU-RIR DATASET	49
3.1. Room Impulse Response Measurement	51
3.1.1. Theoretical Explanation of RIR Measurement	53
3.1.2. Practical Explanation of RIR Measurement	66
3.2. GTU-RIR Data Collection System	70
3.2.1. Mechanical Construction	72
3.2.2. Electrical Construction	81
3.2.3. GTU-RIR Data Presentation	86
3.3. Microphone Quality	88
3.4. Summary	90
4. EVALUATION OF SSIM LOSS FUNCTION	98
4.1. RIR Generation Using StackGAN with SSIM	98
4.2. Experiments and Evaluation	99
4.2.1. A Human-Centric Evaluation of Generated RIRs	107
4.3. Summary	108
5. GEOMETRIC EMBEDDING IMPROVEMENT	109
5.1. Seperating Geometric and Acoustic Models	110

5.1.1.	The MESHTAE Model	112
5.1.2.	The RCRIR Model	116
5.1.3.	The DIMRIR Model	117
5.2.	Experiments and Evaluation	121
5.3.	Summary	123
6.	CONCLUSIONS	124
6.1.	Future Work	126
	REFERENCES	128
	BIOGRAPHY	140
	PUBLICATIONS WITHIN the SCOPE of THESIS STUDY	141
	APPENDICES	CD



LIST of ABBREVIATIONS and ACRONYMS

CNN	:	Convolutional Neural Network
DAS	:	Diffuse Acoustic Simulator
DTFT	:	Discrete-Time Fourier Transform
DNN	:	Deep Neural Networks
DWM	:	Digital Waveguide Mesh
ESS	:	Exponential Sine Sweep
FDTD	:	Finite Difference Time Domain
FFT	:	Fast Fourier Transform
GAN	:	Generative Adversarial Neural Network
GAT	:	Graph Attention Network
GCN	:	Graph Convolutional Network
GDN	:	Graph Deconvolutional Network
GTU	:	Gebze Technical University
HVS	:	Human Visual System
IFFT	:	Inverse FFT
IRS	:	Inverse Repeated Sequence
IR	:	Impulse Response
LTI	:	Linear Time-Invariant
MFCC	:	Mel-Frequency Cepstral Coefficients
MLS	:	Maximum Length Sequence
MSE	:	Mean Squared Error
NERF	:	Neural Radiance Fields
RIR	:	Room Impulse Response
SDN	:	Scattering Delay Network
SNR	:	Signal To Noise Ratio
SSIM	:	Structural Similarity Index Measure
STFT	:	Short Time Fourier Transform
TDS	:	Time Delay Spectrometry
TSP	:	Time-Stretched Pulses
WER	:	Word Error Rate

LIST OF FIGURES

	Page
Figure 1.1: Perception of reverberation.	2
Figure 1.2: Illustration of reverberation of a sound signal.	3
Figure 1.3: RIR generation process.	6
Figure 2.1: MLS sample signal.	14
Figure 2.2: TDS sample signal.	14
Figure 2.3: ESS sample signal.	15
Figure 2.4: Image Method.	16
Figure 2.5: Computational Modeling of Room Acoustics.	17
Figure 2.6: Digital Waveguide Mesh.	18
Figure 2.7: Example Generated Room.	23
Figure 2.8: Overview of training technique of Wager.	24
Figure 2.9: Image2Reverb system architecture.	25
Figure 2.10: TS-RIRGAN architecture.	26
Figure 2.11: FAST-RIR architecture.	26
Figure 2.12: MESH2IR architecture.	27
Figure 2.13: GWA model [56] .	27
Figure 2.14: Illustration of glitches on a sample RIR data.	29
Figure 2.15: SSIM and MSE perofmance image comparison.	30
Figure 2.16: MSSIM behavior for constant images.	33
Figure 2.17: MFCC of a sample RIR.	35
Figure 2.18: Reference RIR signal.	38
Figure 2.19: Distortions of the Mean Contrast Stretch.	39
Figure 2.20: Distortions of the Luminance Shift.	40
Figure 2.21: Various types of distortions.	41
Figure 2.22: Various representations for 3D data.	42
Figure 2.23: Proposed framework of Lin et al.	46
Figure 2.24: Architecture of a Graphormer Encoder.	47
Figure 3.1: BUT ReverbDB data spatial distributions.	51
Figure 3.2: GTU-RIR data spatial distributions.	52
Figure 3.3: RIR measurement in a room.	53
Figure 3.4: LTI System.	54
Figure 3.5: Continuous-Time and Discrete-Time signals.	55
Figure 3.6: Discrete-Time LTI system.	56
Figure 3.7: Decomposition of a discrete-time signal.	57
Figure 3.8: Discrete-Time LTI system principles.	59
Figure 3.9: Convolution example [125].	59
Figure 3.10: Single impulse contains all frequencies [125].	64
Figure 3.11: Discrete Time Fourier Transform. [138]	65
Figure 3.12: Example Inverse Filter [141].	67
Figure 3.13: Example RT60 values for various signals.	70
Figure 3.14: GTU-RIR data collection mechanism.	71
Figure 3.15: GTU-RIR data collection system.	72
Figure 3.16: Microphone and loud speaker directions.	76
Figure 3.17: Step motor mounting.	76
Figure 3.18: Loud speaker stand.	77
Figure 3.19: Loud speaker stand front.	77
Figure 3.20: Loud speaker stand back.	78
Figure 3.21: Microphone stand.	78
Figure 3.22: Microphone stand front.	79
Figure 3.23: Microphone stand back.	79
Figure 3.24: Microphone stand number stickers.	80
Figure 3.25: USB connections.	81
Figure 3.26: Microphone stand.	82
Figure 3.27: Microphone stand cables.	83

Figure 3.28: Loud seaker stand.	84
Figure 3.29: Loud speaker stand.	85
Figure 3.30: Rooms where audio data is collected.	91
Figure 3.31: Frequency spectrum of microphone 1.	92
Figure 3.32: Frequency spectrum of microphone 2.	92
Figure 3.33: Frequency spectrum of microphone 3.	93
Figure 3.34: Frequency spectrum of microphone 4.	93
Figure 3.35: Frequency spectrum of microphone 5.	94
Figure 3.36: Frequency spectrum of microphone 6.	94
Figure 3.37: The maximum MSE coherence plot.	95
Figure 3.38: The minimum MSE coherence plot.	95
Figure 3.39: The maximum SSIM coherence plot.	96
Figure 3.40: The minimum MSE coherence plot.	96
Figure 3.41: The max. GLITCH COUNT coherence plot.	97
Figure 3.42: The min. GLITCH COUNT coherence plot.	97
Figure 4.1: FAST-RIR/MESH2IR generated/real RIRs.	101
Figure 4.2: Allignment of waves.	102
Figure 4.3: Coordinate systems conversion.	102
Figure 4.4: GTU-RIR coherence plots.	105
Figure 4.5: BUT ReverbDB coherence plots.	106
Figure 5.1: Example ray-casted floor plan image.	110
Figure 5.2: Mesh transformer auto-encoder architecture.	111
Figure 5.3: Floor plan of a sample apartment flat.	113
Figure 5.4: A 3D sample in synthetic apartment flat dataset.	114
Figure 5.5: MESH-TRANSFORMER-AE best mesh.	114
Figure 5.6: MESH-TRANSFORMER-AE average mesh.	115
Figure 5.7: MESH-TRANSFORMER-AE worst mesh.	115
Figure 5.8: Ray casting process.	116
Figure 5.9: RCRIR geometric model.	117
Figure 5.10: RCRIR model.	118
Figure 5.11: DIMRIR Model.	119
Figure 5.12: RIRs generated by MESHTAE and MESH2IR.	119

LIST OF TABLES

	Page
Table 2.1: Recommended Reverberation Time Table [36].	9
Table 2.2: RIR Measurement/Generation Comparison Table [6].	10
Table 2.3: MSE and SSIM values for various distortion types.	37
Table 3.1: Public RIR Datasets [120] [121]	49
Table 3.2: Sound absorption coefficients.	67
Table 3.3: Bill of materials.	73
Table 3.4: GTU-RIR BUT-ReverbDB data statistics comparison.	87
Table 3.5: RIR Comparison per microphone.	89
Table 4.1: Comparison of loss values based on SSIM and MSE.	104
Table 4.2: Results of survey on human perception quality.	108
Table 5.1: Model comparison table.	120

1. INTRODUCTION

Far-field automatic speech recognition applications, such as virtual home assistants, require either dereverberating the received sound signal or training the recognizer neural network with reverberated sound data based on the given environment conditions. Similarly, virtual and augmented reality applications need to auralize sound signals for a specific virtual environment, considering the sound source and receiver locations, to provide a natural auditory experience. Additionally, applications like room size and shape estimation, direction of arrival estimation, material detection, listener envelopment evaluation in concert halls, ultrasound tomography, and water pipeline defect detection in buildings using sound signals rely on the analysis of reverberated sound signals.

The analysis of reverberated sound signals is primarily conducted using room impulse response (RIR) signals, which are either measured or generated for a given environment. The Motivation section explains the concepts of “reverberation” and “room impulse response,” discusses RIR measurement and generation techniques, and highlights the motivation for researching RIR generation using neural networks. The Problem Definition section clearly defines the problem and outlines the steps taken to address it. Finally, the Contributions section summarizes the key contributions of this thesis.

1.1. Motivations

The sound experienced inside a room is a combination of the direct sound, which travels directly from the source to the listener, and reflected sound, which bounces off various surfaces within the room. These reflected sound waves may undergo multiple interactions, such as additional reflections, refractions, or diffractions, before reaching the listener. Architectural elements of the room, including walls, floors, ceilings, furniture, and even people, are the primary sources of these reflections [1]. While a single reflection is perceived as an echo, multiple overlapping reflections form reverberation, which is often perceived as noise [2]. This phenomenon is visually illustrated in Figure 1.1, while Figure 1.2 depicts the representation of reverberation in a recorded sound signal.

Reverberation, commonly known as reverb, is the persistence of sound after the original sound source has stopped emitting it. This phenomenon occurs when sound waves reflect off surfaces within an environment, creating multiple overlapping reflections that gradually decay as their energy is absorbed by surrounding materials such as furniture, people, and air. Reverberation is most noticeable when the sound source ceases, but the reflections continue, gradually decreasing in amplitude until they become imperceptible [4]. The reflection, refraction, and diffraction of sound waves do not alter their frequency, as frequency depends solely on the source rather than the medium. The only impact on

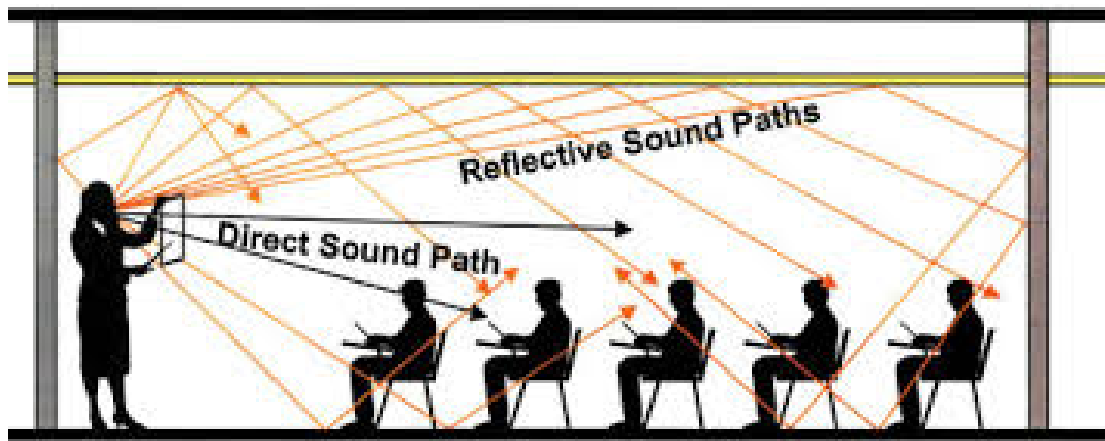


Figure 1.1: Perception of reverberation from different locations in the room [3].

frequency is that certain sound waves at specific frequencies are absorbed by particular materials within the environment [5].

The reverberation properties depend on several factors, including:

- Room Geometry: Shape and size of the room and the shapes and positions of the objects in the room.
- Sound Source and Receiver Positions: Placement of the loudspeaker and microphone.
- Reflection Coefficients of Materials: Acoustic properties of the surfaces in the room.
- Environmental Conditions: The sound propagation in the room changes with the temperature and humidity change. Also the objects in the room changes sound reflection coefficients with the temperature.

Room Impulse Response (RIR) measurement is an audio signal processing task aimed at capturing and analyzing the acoustic characteristics of a room or environment. The goal is to measure and model how sound waves interact with the space, including reflections, reverberation, and echoes. However, measuring the RIR can be challenging or even impossible in certain scenarios—such as when the environment is virtual (e.g., in virtual reality applications), difficult to access (e.g., inside a welded metal enclosure), or restricted (e.g., in damaged buildings). In such cases, generating RIRs becomes necessary using the room's geometry, the reflection coefficients of materials within the space, and the positions of the loudspeaker and microphone [6].

In virtual reality applications, such as games or simulations, there is no physical environment in which to measure the RIR. To achieve immersive auralization, we must

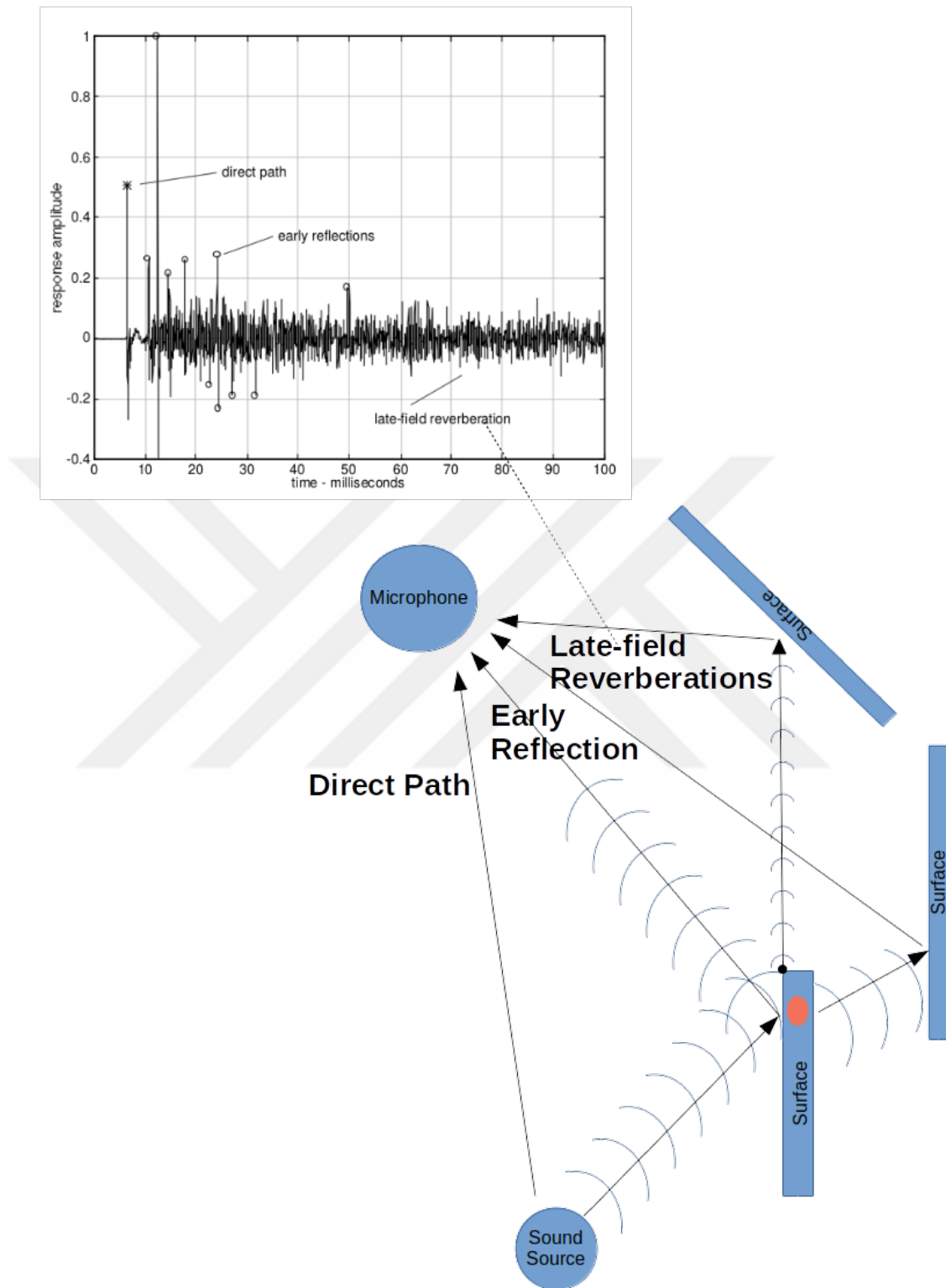


Figure 1.2: Illustration of reverberation of a sound signal [3].

generate reverberated sound artificially. This is done by generating an RIR based on the virtual room's characteristics and then convolving it with clean sound [7].

Similarly, in physically inaccessible spaces, RIR measurement becomes challenging and costly. For instance, consider a welded metal object where all joints are sealed, but its mesh geometry file is available. If we can estimate the RIR inside such an object with reasonable accuracy, we may be able to detect structural defects or breakages [8].

Another example is a damaged building. If we have access to the building’s original plan (i.e., its mesh geometry), we can estimate its expected RIR. By comparing this generated RIR with an RIR measured in the building’s current damaged state, we can roughly infer which parts of the structure have been affected [9–11].

Apart from these examples, RIR generation finds applications in various fields. It is utilized for estimating a room’s size and shape [9–11], enhancing augmented reality audio experiences [7], and determining the direction of incoming sound [12]. Moreover, it aids in identifying room materials [13], detecting the locations of reflecting surfaces [14], and conducting ultrasound tomography [15]. Another significant application is detecting defects in water pipelines within buildings [8]. By simulating RIRs for specific environments, these applications achieve greater accuracy and usability in their respective domains.

Several techniques have been proposed for generating Room Impulse Responses (RIRs), including ray-based, wave-based, statistical, and neural network-based methods. Ray-based approaches, such as the image method [16], estimate the RIR by modeling sound reflections within a room. Wave-based techniques, including the Discrete Wavelet Method (DWM) [17, 18] and Finite Difference Time Domain (FDTD) [19], compute sound propagation by solving wave equations. In contrast, statistical models [20–22] approximate room acoustics based on statistical properties, as demonstrated in the works of Erkelens et al. and Florencio et al.

Among these approaches, neural network-based methods [23–30] have gained popularity due to their superior trade-off between computational complexity and quality. These methods leverage deep learning techniques to generate RIRs, offering an efficient alternative to traditional methods. For instance, models like MESH2IR and FAST-RIR have been shown to deliver high-quality RIRs while reducing computational costs. Neural network-based methods, including GANs, have become widely adopted for this task, owing to their ability to model complex relationships between room geometry, acoustic features, and environmental variables without requiring explicit physical simulations.

This thesis focuses on Room Impulse Response (RIR) generation using deep learning techniques, exploring methods to create more realistic RIRs and ways to decouple geometric and acoustic models to enhance the modularity of RIR generation. The motivation behind this work stems from the need to generate accurate RIRs in various scenarios, particularly for environments where physical measurements are not possible,

such as virtual environments or inaccessible real rooms. Traditional methods for RIR generation often face limitations in flexibility and accuracy, which deep learning-based approaches aim to overcome.

1.2. Problem Definition

Given the room geometry, along with the microphone and speaker locations, room impulse response (RIR) generator models aim to predict the impulse response sound signal with the highest possible accuracy. Most of the RIR generators currently ignore the reflection coefficients of the materials in the room because it is very hard to obtain the material knowledge and their sound reflection coefficients. Only two of the papers take the reflection coefficients into account either by acquiring the material reflection information as node feature in the 3D mesh data [31] or by acquiring the photograph of the room [32]. These methods are not currently very efficient, so we don't take the materials' sound reflection coefficients into account in this thesis for RIR generation. This thesis focuses on the use of Generative Adversarial Networks (GANs) for acoustic modeling and transformers as well as graph neural networks (GNNs) for embedding room geometry, enabling a modular and efficient approach to RIR generation.

The RIR generation process, as outlined in Figure 1.3, follows these steps:

- **Acquire Real RIR Data:** Collect real-world RIR data for training and benchmarking the model.
- **Embed the Room Geometry:** Process the room's geometrical data to generate embeddings using transformer or GNN-based methods.
- **Generate Room Impulse Response:** Use the GAN model to synthesize RIRs based on the geometric embeddings and input parameters.
- **Evaluate Generated RIRs:** Compare the generated RIRs to real RIRs using metrics such as Mean Squared Error (MSE) and Structural Similarity Index Measure (SSIM) to assess accuracy and quality.

This methodology addresses key challenges in the field while leveraging advanced deep learning techniques to achieve realistic and high-quality RIR generation.

Real RIR data is crucial for testing RIR generator models as it directly determines their accuracy. To address this need, we developed an automated system capable of collecting RIR data at various points within a room. This system uses stepper motors and a mechanical rotating mechanism, with speakers and microphones mounted to ensure precise data acquisition. During the course of this PhD study, we designed and implemented this RIR collection system, conducted data collection, and documented every aspect of the process. The documentation includes the bill of materials, mechanical

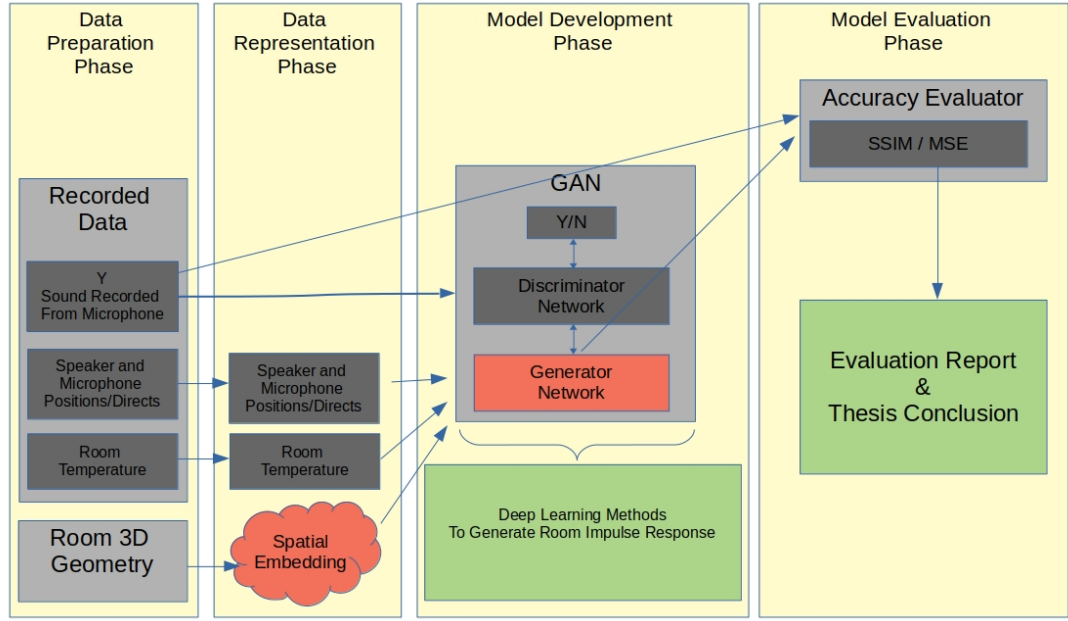


Figure 1.3: RIR generation process.

and electrical designs, electronic configurations, operating system setups, Python scripts for automation, and the collected dataset itself. All these materials have been made publicly available in the GTU-RIR GitHub repository [33] to facilitate reproducibility and support further advancements in the field.

RIR generator models frequently produce glitches in the generated RIRs, causing distortions in the resulting reverberated signals. To address this issue, we evaluated the impact of different loss functions on reducing these glitches. An algorithm was first developed to detect glitches effectively. Subsequently, two GAN models were chosen, and their generator loss functions were modified to incorporate various metrics. These models were trained with the updated loss functions and evaluated using two real RIR datasets, including the GTU-RIR dataset. The loss functions tested included SSIM, MSE, SSIM of MFCCs, and combinations of these metrics, enabling a comprehensive assessment of their effectiveness in improving the quality and consistency of the generated RIRs.

Another challenge with current RIR generator models is the coupling of geometric and acoustic neural network components, requiring them to be trained together using RIR data. This approach makes it challenging to generate RIRs for different types of geometries, as the entire model must be retrained for each new geometry type. To address this, we modified the RIR generator model, separating the training process into two distinct stages. This decoupling necessitated replacing the geometric component, initially a Graph Convolutional Neural Network, with a Mesh Transformer Autoencoder.

To pre-train the Mesh Transformer Autoencoder, we created a synthetic mesh dataset using Python trimesh scripts, as documented in the GTU-RIR repository [33].

By achieving these objectives, our research seeks to advance the field of RIR generation by providing a standardized framework for collecting real RIR data using accessible commodity hardware, reducing glitches in generated RIRs, and decoupling geometric and acoustic models to enhance the modularity and flexibility of RIR generator neural networks.

1.3. Contributions

Our contribution to literature are as follows:

- GTU-RIR Dataset :
 - We introduce a new real RIR dataset that offers more spatially distributed data points than existing datasets, with a higher quantity of data points.
 - A comprehensive explanation of the data collection system, including hardware specifications and scripts, is provided in the "gtu-rir" GitHub repository [33].
 - Our RIR collection system can also be utilized for the acoustic evaluation of buildings. In particular, reverberation time values are crucial for different room types, as shown in Table 2.1.
- Decreasing RIR glitches using SSIM loss function :
 - Various loss functions, including MFCC-based SSIM, SSIM, MSE, and combinations thereof, were evaluated.
 - Two different GAN models and two distinct real RIR datasets were utilized to evaluate the loss functions, ensuring that the proposed loss function performs effectively across varying models and datasets.
 - We concluded that mixture of SSIM and MSE as loss function gives the best results.
- Separating geometric and acoustic models of RIR generating GANS :
 - A synthetic apartment flat mesh dataset was created using the Python Trimesh library, with scripts available in the "gtu-rir" GitHub repository.
 - We offered a new mesh embedding neural network based on Transformer Autoencoders, and pre-trained this neural network using the synthetic dataset we produced.
 - We developed a new mesh embedding neural network based on Transformer

Autoencoders and pre-trained this network using the synthetic dataset.

- The graph neural network-based geometric embedding model of the MESH2IR neural network was replaced with the Mesh Transformer Autoencoder, resulting in the new MESH-TAE model.
- This approach allows separate training of the geometric model and the acoustic GAN model. Consequently, our method enables adapting the RIR generator neural network to new geometries without retraining the acoustic GAN component.

1.4. Outline

The structure of the thesis is as follows:

Chapter 1 provides an introduction, outlining the motivation behind the study and its contributions. Chapter 2 offers a literature review on RIR generation techniques, the use of the SSIM loss function, mesh transformers, and neural networks for geometric embedding. Chapter 3 presents existing RIR datasets and introduces the new GTU-RIR dataset, along with detailed descriptions of the RIR collection system, including its hardware and configuration. Chapter 4 addresses the problem of glitches in RIRs generated by GANs, proposing a solution using the SSIM loss function, and presents experimental results using real datasets like GTU-RIR and BUT ReverbDB. Chapter 5 introduces a novel method for geometric embedding, enabling the separation of geometric and acoustic models in RIR generation. Finally, Chapter 6 concludes the thesis, summarizing the key findings and offering suggestions for future research directions.

2. LITERATURE REVIEW

Reverberation time is a metric used to quantify the duration it takes for sound to diminish to inaudibility in an enclosed space after the sound source ceases [4]. Wallace Clement Sabine (1868–1919), a pioneer in architectural acoustics, studied the acoustic properties of rooms and discovered a correlation between the acoustic quality, the room’s volume, and the absorption characteristics of its surfaces [34]. He formulated an empirical relationship to describe this phenomenon using the organ pipes at the seven octave frequencies 64 Hz to 4096 Hz, corresponding to the music tones C [35].

Sabine’s equation links the reverberation time for a 60 dB decay, denoted as RT_{60} , with the room’s volume (V) and its total absorption (S.a), where:

- C_{20} : Speed of sound in air at $20^{\circ}C$
- V: Volume of the room in m^3
- S: Total surface area of the room in m^2
- a: Average absorption coefficient of the room surfaces.
- S.a: Total absorption in sabins.

This relationship is mathematically expressed in Equation 2.1, serving as a fundamental principle for predicting and optimizing acoustic properties in architectural design. Reverberation time is a crucial parameter for human auditory perception. For example, a classroom should have a very short reverberation time, whereas a concert hall requires a longer reverberation time to enhance musical quality, as shown in Table 2.1. These values represent the average across frequencies ranging from 0.1 kHz to 5 kHz, while the typical human hearing range extends from 0.1 kHz to 20 kHz. Reverberation time varies across different frequency bands due to the frequency-dependent nature of material reflection coefficients [36].

$$RT_{60} = \frac{24ln10^1}{C_{20}} \frac{V}{S.a} \quad (2.1)$$

Table 2.1: Recommended Reverberation Time Table [36].

Location	Volume	Recommended Reverberation Time
Recording Studio	< 50 m ³	0.3 s

Table 2.1: (continued) Recommended Reverberation Time Table [36].

Location	Volume	Recommended Reverberation Time
Classroom	< 200 m ³	0.4 - 0.6 s
Office, Conference Hall	< 1000 m ³	0.5 - 1.1 s
Lecture Hall	< 5000 m ³	1.0 - 1.5 s
Concert Hall, Opera	< 20000 m ³	1.4 - 2.0 s
Church		2 - 10 s

Since Sabine's foundational work, researchers have explored various methods to measure and simulate the sound perceived in a room, aiming to capture the complexity of acoustic interactions within enclosed spaces. These methods span a range of approaches, from direct measurement techniques to advanced computational simulations, each offering unique advantages and limitations.

Christensen [6] explored the most effective methods for obtaining room impulse responses (RIRs), comparing the advantages of measuring RIRs versus simulating (generating) them. The key facts related to both measurement and simulation are summarized in Table 2.2. From the comparison in Table 2.2, it is clear that once an accurate RIR generation system is established, RIRs can be generated for any room geometry without the need for physical presence or dealing with issues like background noise and distortion.

Moreover, there are scenarios where generating RIRs is the only viable option, such as in virtual reality environments where physical measurements are not feasible. Additionally, the RIR generation process could be adapted for 3D geometry reconstruction by reversing the inputs, outputs, and neural network architecture. This adaptation has potential applications in medical devices, such as ultrasound tomography, where accurate modeling of room acoustics and geometry is essential.

Table 2.2: RIR Measurement/Generation Comparison Table [6].

FACT	RIR MEASUREMENT	RIR GENERATION
Room geometry	Fully included by definition	Approximated
Alteration of room geometry	Difficult	Easy

Table 2.2: (continued) RIR Measurement/Generation Comparison Table [6].

FACT	RIR MEASUREMENT	RIR GENERATION
Wave phenomena (phase information, diffraction)	Fully included – inherent in the real sound field	Approximated with varying accuracy
Wall properties	Fully included – inherent in the real room	Absorption - scattering coefficients have to be measured or assumed, with limited accuracy
Air absorption (a function of temperature and humidity)	Fully included but may vary significantly in different measurements	Calculated from theoretical formulas, but very accurate
Source directivity	Not perfect: Lobes at high frequencies	Perfectly omni-directional
Dynamic range of source	Insufficient at very low and very high frequencies Distortion at high levels	Unlimited dynamic range at all frequencies No distortion
Calibration of source	Special procedure needed for the G parameter	Perfect per definition
Background Noise	Present and limits the dynamic range	Not present
Microphone directivity	Omnidirectional microphone Some parameters require figure-of eight pattern or a dummy head	All directivities available
Results in different octave-bands	Filtering is required, which alters the original signal	Results are derived directly in different bands - no alteration due to filtering.
Reproducibility	Not perfect: Depends heavily on the source	Can be perfect, depending on the algorithm
Influence of operator	Knowledge and experience important	Knowledge and experience very important

In the following sections, we review recent studies and developments in Room Impulse Response (RIR) measurement and generation, emphasizing the advancements in methodologies, tools, and algorithms that have emerged over time.

2.1. Room Impulse Response Measurement

Room impulse responses (RIRs) are measured to capture the acoustic characteristics of a space, serving as its unique acoustical fingerprint. These RIRs can then be applied in various ways, such as enhancing a room's acoustical quality, simulating reverberated sound for that environment, or removing reverberation from audio recordings to restore clarity. Once the RIR for a specific location in a room is obtained, it becomes possible to process any input sound signal to mimic the acoustic effect of being at that specific location.

The choice of measurement techniques significantly influences the quality of the RIR signal, which in turn affects the fidelity and effectiveness of applications relying on these measurements, such as virtual simulations or noise reduction. In this section, we review studies and advancements in RIR measurement techniques, emphasizing their methodologies, applications, and impact on signal quality.

The rationale behind measuring an impulse response rather than using an ordinary sound signal lies in achieving a higher signal-to-noise ratio (SNR). Impulse signals allow for clearer differentiation of the acoustic characteristics of the room by minimizing the effects of noise. To further reduce noise during RIR measurements, recordings are typically conducted during the quietest hours in the room. In this process, the room, sound source, and receiver are treated as a unified system, capturing the acoustic interaction among these elements [37].

Efforts to measure room impulse responses date back to the late 1930s. A pioneering work by Fazenda in 1939, titled "Modulation Transfer Function as a Measure of Room Low Frequency," explored RIR measurements for evaluating room acoustics at low frequencies [38]. This laid the foundation for systematic studies of RIR measurement techniques.

A significant breakthrough occurred in 2000, when Farina et al. introduced the use of exponential sine sweep (ESS) signals for acoustic measurements in their work, "Acoustic Measurements in Opera Houses: Comparison Between Different Techniques and Equipment" [39]. They demonstrated that ESS signals provided superior SNR compared to other signals, making it a preferred choice for precise and reliable RIR measurement.

Stan et al. (2002) compared four impulse signal techniques for measuring room impulse responses: Maximum Length Sequence (MLS), Inverse Repeated Sequence (IRS), Time-Stretched Pulses (TSP), and Exponential Sine Sweep (ESS). They demonstrated that MLS and IRS work better in environments with non-white noise, while ESS is more effective in quiet settings due to its superior signal-to-noise ratio.

Picinali (2006) provided an extensive overview of methods for extracting impulse responses, including the use of MLS and ESS signals. He discussed their relationship to linear time-invariant (LTI) systems and techniques such as convolution, deconvolution, and their relevance in head-related transfer functions (HRTFs).

Holters et al. (2009) investigated the practical applicability of MLS, Time Delay Spectrometry (TDS), and ESS in real-world scenarios. They presented the advantages and limitations of each method, offering guidance for choosing the appropriate algorithm. Key experimental findings include:

Maximum Length Sequence (MLS): An MLS is a periodic, pseudo-random two-level signal used to extract impulse responses via deconvolution. Its advantages include low computational demands and efficient cross-correlation. However, MLS is highly sensitive to non-linear distortions, requiring careful amplitude management in analog signal paths. An example for this signal may be seen in Figure 2.1 which is expressed using the Equation 2.2.

Time Delay Spectrometry (TDS): TDS uses a linearly swept sine wave (chirp) as the input signal, represented as $x(t) = \sin(k.t^2)$, where k is the sweep rate. Introduced by Heyser in 1967 [40], it eliminates the need for Fourier transforms for transfer function measurements, which was a significant advantage at the time. However, it was initially limited to measuring only short impulse responses. An example for this signal may be seen in Figure 2.2.

Exponential Sine Sweep (ESS): ESS handles non-linearities gracefully, enabling higher measurement amplitudes and reduced time for achieving desired accuracy. It also provides better accuracy at lower frequencies, aligning well with human auditory perception. However, it requires more computational resources compared to other techniques. An example for this signal may be seen in Figure 2.3 which is expressed using the Equation 2.3.

Each method has strengths suited to specific scenarios, offering a range of tools for precise RIR measurement under various conditions.

$$\begin{aligned}
 a_3[n + 1] &= a_0[n] + a_1[n] \\
 a_2[n + 1] &= a_3[n] \\
 a_1[n + 1] &= a_2[n] \\
 a_0[n + 1] &= a_1[n]
 \end{aligned} \tag{2.2}$$

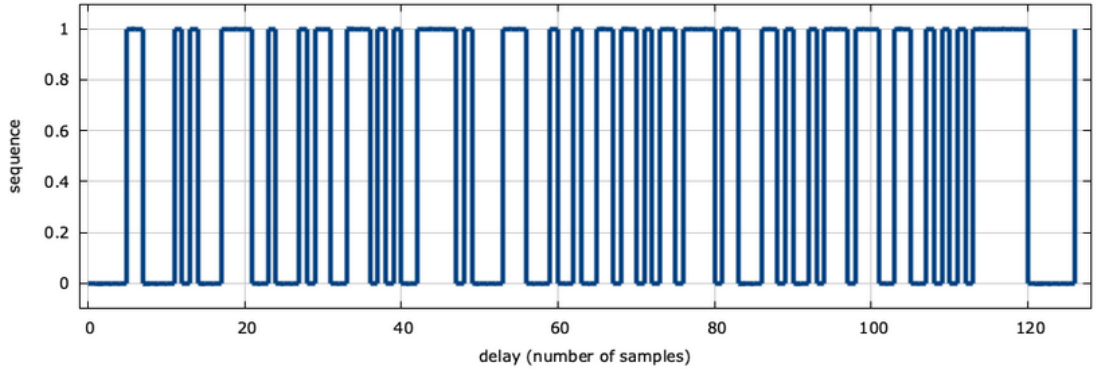


Figure 2.1: Maximum length sequence sample signal [41].

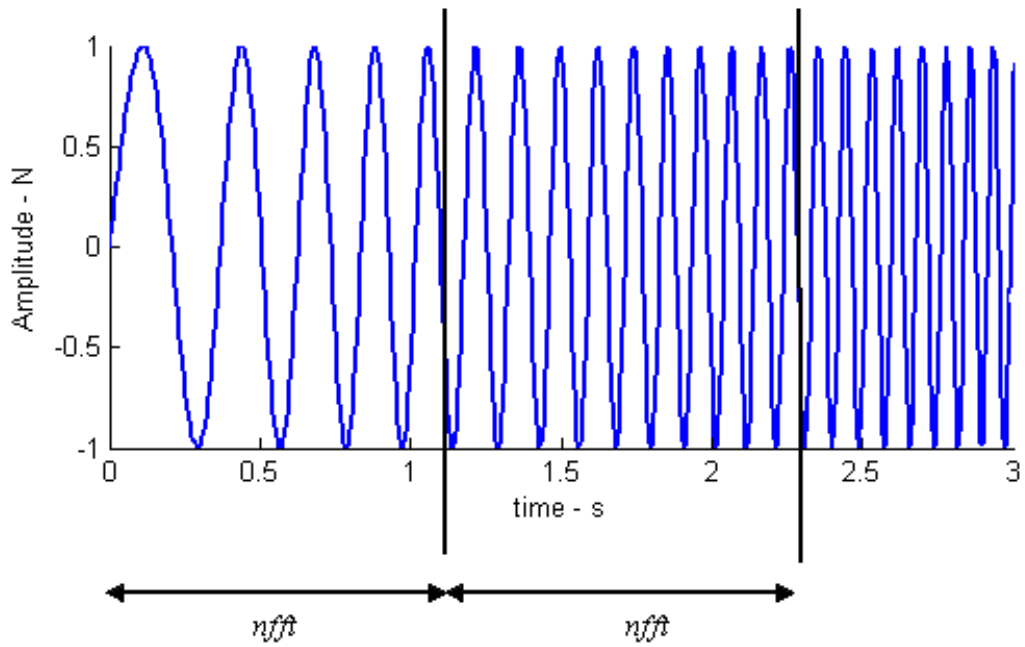


Figure 2.2: Time delay spectrometry (linear sine sweep) sample signal [42].

$$x(t) = \sin\left(\frac{2\pi f_1 T}{R} \cdot (e^{\frac{tR}{T}} - 1)\right)$$

$$R = \ln\left(\frac{f_2}{f_1}\right) \quad (2.3)$$

Canfield-Dafilou et al. (2018) [44] introduced a Portable Impulse Response Measurement System (PRIMS), designed to enable artists and researchers to capture high-quality impulse responses in challenging and remote locations. This system is compact enough

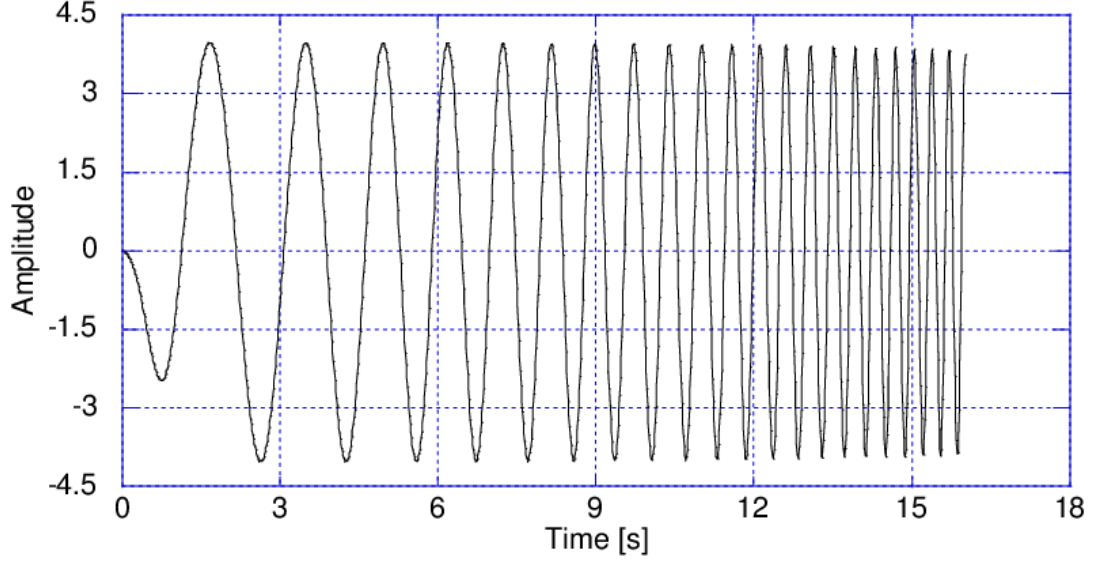


Figure 2.3: Exponential sine sweep sample signal [43].

to fit in a backpack and lightweight for transportation through difficult terrains while maintaining high audio quality.

Building on insights from previous works, including PRIMS and studies comparing RIR measurement methods, we opted to use the Exponential Sine Sweep (ESS) signal for generating the GTU-RIR dataset. ESS was selected for its superior handling of non-linearities and better signal-to-noise ratio, particularly in quiet environments.

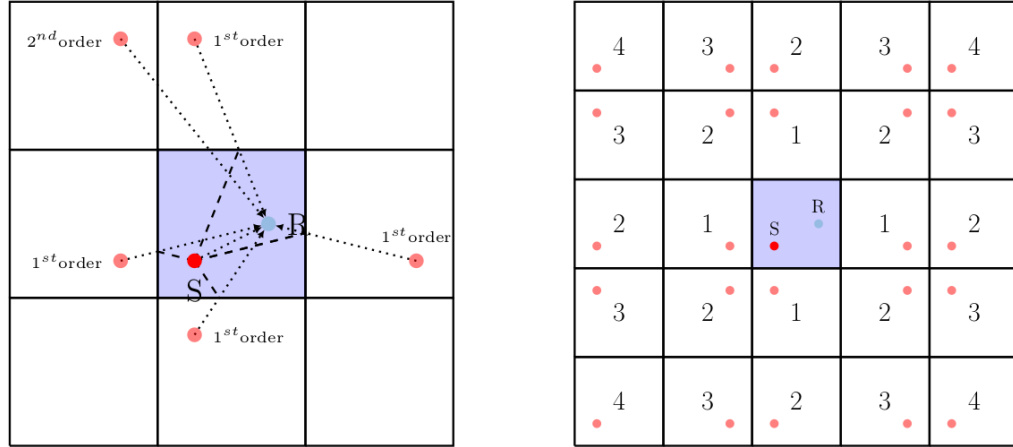
To ensure practicality and high-quality recordings, we designed an easily transportable microphone and speaker system. Room impulse responses were recorded during night-time to minimize noise interference. Additionally, we included 4 seconds of pre- and post-recording silence in each capture to facilitate noise cleaning during post-processing. This approach ensured robust and reliable RIR data collection while accommodating the practical constraints of real-world environments.

2.2. Room Impulse Response Generation

Computational modeling of room impulse responses began in 1979 with the pioneering paper “Image Method for Efficiently Simulating Small-Room Acoustics” by Jont B. Allen and David A. Berkley [16] from the Acoustic Department of Bell Labs. This seminal work introduced the Image Method, a computational approach where the walls of a room are treated as mirrors for sound, creating virtual images of the sound source that simulate reflections.

Although groundbreaking, their method had significant limitations. It was applicable

only to rectangular, empty rooms with mono-type surfaces. Additionally, the positions of the speaker and microphone were fixed, and the calculations assumed constant environmental conditions, such as temperature, humidity, and pressure. The accuracy of the simulation depended on the reflection order—the maximum number of reflections the system could computationally handle. As the reflection order increased, the modeled reverberation more closely approximated real-world values -.



(a) The figure shows the positions of the source, S, and receiver, R, and some paths related to the positions.

(b) The figure shows the mirror image pattern that corresponds to the initial source position and the reflection orders.

Figure 2.4: Image Method [45].

Room Impulse Response (RIR) signals can be computationally generated using various approaches, categorized as ray-based methods, wave-based methods, or statistical methods.

Ray-based methods, such as the Image Method [16], simulate sound propagation by treating surfaces as mirrors and calculating sound reflections as if light rays were bouncing off them. These methods are computationally efficient but are limited in accuracy when modeling complex geometries or diffraction effects.

Wave-based methods, such as the Finite-Difference Time-Domain (FDTD) Method [19], solve wave equations directly on a discretized space-time grid, capturing fine details of wave behavior, including diffraction and interference. While highly accurate, these methods are computationally intensive and require significant memory, particularly for high frequencies or large spaces.

Statistical methods involve techniques like neural-network-based models or the statistical estimation of RIR signal parameters. These methods leverage data-driven or probabilistic

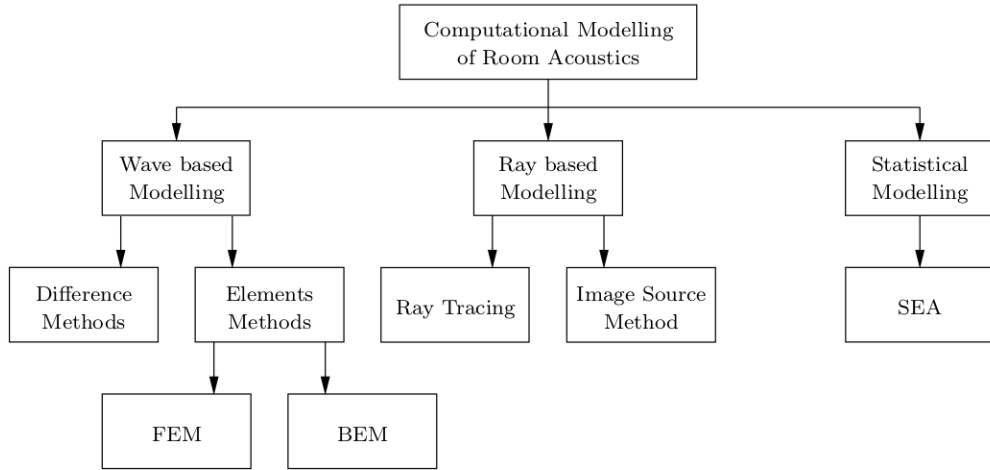


Figure 2.5: Computational Modeling of Room Acoustics [45].

models to estimate RIR signals, offering flexibility and generalizability across different scenarios.

A categorization of these methods is depicted in Figure 2.5. Regardless of the approach, these methods require inputs such as room geometry, loudspeaker positions, and microphone locations to produce the desired RIR signal as output.

2.2.1. Ray Based Methods

The Image Method [16] simulates sound reflections by tiling copies of the room geometry and placing "mirror sources" within each tile to represent reflections, as illustrated in Figure 2.4. This method is based on several assumptions:

- The room has a rectangular, empty, and mono-type surface geometry.
- There is a single speaker and microphone, both at fixed coordinates.
- Atmospheric conditions (e.g., temperature, pressure, humidity) are constant.
- Sound propagates in straight lines at a uniform speed.
- The energy of the sound decreases proportionally to $1/r^2$, where r is the distance traveled.
- Sound is perfectly reflected at the boundaries, with amplitude reductions proportional to the absorption coefficient of the surface, independent of frequency.

An implementation of the Image Method was provided by Habets [45]. While computationally efficient, this ray-based approach simplifies sound as light-like beams, ignoring

its wave nature, which can lead to inaccuracies, particularly at low frequencies where wave effects such as diffraction and interference dominate.

Alternative wave-based methods (e.g., the Finite Element Method (FEM) or the Finite-Difference Time-Domain (FDTD) approach) and statistical methods (e.g., Statistical Energy Analysis (SEA)) address these limitations to some extent. However, they, too, operate under simplifying assumptions. Non-deep-learning methods like these often rely on idealized conditions and face challenges in accurately modeling complex geometries or heterogeneous materials.

Ray-based methods like the Image Method excel in efficiency but produce poor-quality RIRs at low frequencies due to their inability to account for sound's wave properties, highlighting the limitations of traditional computational models.

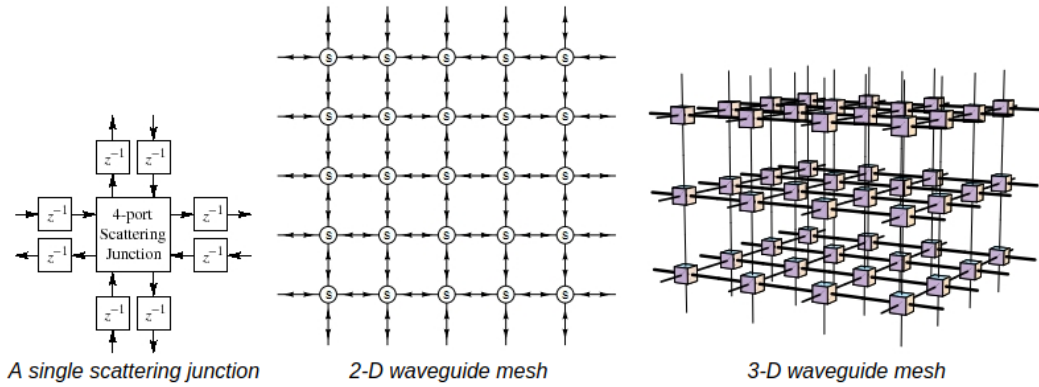


Figure 2.6: Digital Waveguide Mesh [17].

2.2.2. Wave Based Methods

Wave-based methods are known for producing high-quality room impulse response (RIR) signals due to their ability to accurately capture the wave behavior of sound, including diffraction, interference, and other complex phenomena. However, their high computational complexity often limits their practicality for real-time applications. Despite this, the precision of wave-based methods makes them valuable tools for generating synthetic RIR data, which can be used for training and validation purposes in various acoustic studies.

One of the most widely used wave-based methods in acoustic simulations is the Digital Waveguide Mesh (DWM) [46]. The DWM models sound propagation by discretizing the wave equation on a spatial grid, allowing for detailed simulation of sound waves in

a room.

The first application of DWM in room acoustics simulation was demonstrated by Savioja et al. in 1994 [47], establishing its utility for this purpose. Subsequently, in 2007, Damian et al. employed the DWM method specifically for generating RIRs [17], leveraging its high accuracy to model room acoustics with precision.

Although computationally expensive, the DWM and other wave-based approaches remain critical for tasks requiring highly accurate synthetic RIR data, serving as benchmarks for testing new RIR generation methods and acoustic models.

The Digital Waveguide Mesh (DWM) is a numerical simulation technique that employs a spatially discretized grid to model acoustic wave propagation within a given domain. This approach is particularly well-suited for simulating the behavior of vibrating objects that support sound wave transmission, as depicted in Figure 2.6. By solving the wave equation on a mesh, the DWM method provides a detailed representation of sound propagation and resonance.

In three-dimensional simulations, the DWM's computational requirements become exceptionally high, making it impractical for scenarios demanding real-time processing. Beyond the general computational burden, the method faces a significant limitation in the form of dispersion error. Dispersion error occurs when the wave velocity depends on the frequency and direction of propagation, causing inaccuracies in wave behavior. This leads to propagation errors and mistuning of the resonant modes, which are particularly problematic when high accuracy is required.

The degree of dispersion error is influenced by the mesh topology used in the simulation. Different mesh configurations exhibit varying levels of error, with some topologies offering better accuracy at the cost of increased computational complexity. Consequently, careful selection and optimization of the mesh are critical to minimizing dispersion error and enhancing the fidelity of DWM-based acoustic simulations.

Fundamentally, synthesizing the characteristics of a bounded space using a DWM is exactly the same as synthesizing the sound of a vibrating physical object. However, in the latter example, sound output is generated directly from the modeled object by reading sample values at a scattering junction, whereas with room acoustics modeling it is a room impulse response (RIR) that is synthesized rather than the actual sound source. It has been shown that DWMs offer accurate RIR synthesis at low frequencies and demonstrate natural wave phenomena such as interference and diffraction with high-frequency accuracy being limited by update frequency and the dispersion error of the selected topology. The 2-D digital waveguide mesh formulation, while corresponding exactly with the standard difference scheme approximation, may be implemented in a fully parallel, multiply-free formulation.

In 2009 Hacıhabiboğlu et al. published their work on simulation of directional microphones in Digital-Wave Guide-Mesh based models of room acoustics [18]. This article presents a method for the simulation of directional microphones in DWM-based models of room acoustics. The method is based on the directional weighting of the microphone response according to the instantaneous direction of incidence at a given point. The direction of incidence is obtained from instantaneous intensity that is calculated from local pressure values in the DWM model.

Saarelma and Greco visualized the sound field using the Finite Difference Time Domain (FDTD) method and measured spatial RIR (combined both of the methods for visualization) in 2015 [19]. FDTD solves Maxwell's equations in the time domain using finite-difference approximations. Being a time-domain method, FDTD is more intuitive than other techniques and works by creating a "movie" of the fields flowing through a device.

Synthesizing the acoustical properties of a bounded space using a Digital Waveguide Mesh (DWM) parallels the process of synthesizing sound from a vibrating physical object. In both cases, the underlying method relies on solving wave equations within a discrete spatial framework. The key difference lies in the output: for vibrating objects, the output is sound sampled directly at specific scattering junctions, while for room acoustics, the goal is to synthesize a Room Impulse Response (RIR).

DWM Characteristics for Room Acoustics:

- **Low-Frequency Accuracy:** DWMs are particularly effective at synthesizing RIRs at lower frequencies, where natural wave behaviors such as interference and diffraction are well-represented.
- **High-Frequency Limitations:** The accuracy at higher frequencies is constrained by the update frequency and the dispersion error introduced by the mesh topology. Dispersion error affects wave propagation speed and results in phase inaccuracies.
- **Efficient Implementation:** A 2D DWM formulation can be implemented in a fully parallel, multiply-free configuration, making it computationally efficient for certain applications while still offering fidelity.

Notable Applications and Advancements:

- **Directional Microphones in DWM Models (Hacıhabiboğlu et al., 2009) [18]:**
 - Proposed a method to simulate directional microphones in DWM-based room acoustics models.
 - The approach uses directional weighting to adjust the microphone response based on the instantaneous direction of sound incidence.
 - Direction of incidence is derived from instantaneous intensity, calculated

using local pressure values within the DWM model. This allows for more realistic simulation of microphone behavior in complex acoustic environments.

- Spatial Visualization using FDTD (Saarelma & Greco, 2015) [19]:
 - Combined the Finite Difference Time Domain (FDTD) method with spatial RIR measurements for visualizing sound fields.
 - FDTD, a time-domain simulation technique, solves wave equations using finite-difference approximations and provides an intuitive representation of sound field evolution.
 - This approach facilitates the creation of a "movie" that visually captures sound propagation and interactions in the modeled space, enhancing the understanding of spatial acoustics.

These advancements highlight the versatility of wave-based methods like DWM and FDTD in simulating room acoustics. While DWM excels in synthesizing RIRs with natural wave phenomena, FDTD provides a robust framework for visualizing sound fields in time and space.

Atalay et al. (2022) introduced a Scattering Delay Network (SDN) simulator for modeling coupled-volume acoustics in their work titled "Scattering Delay Network Simulator of Coupled Volume Acoustics" [48]. The SDN is a type of digital waveguide network (DWN) designed to balance physical accuracy and computational efficiency, making it suitable for real-time interactive simulations of room acoustics.

The SDN enables direct parametric control of the geometry of a simulated cuboid enclosure as well as the directional characteristics of sound sources and microphones. This feature allows for flexible and detailed acoustic simulations. The computational design of the SDN ensures low resource consumption, making it possible to achieve interactive performance, which is especially useful for applications such as virtual reality and live acoustic evaluations.

The research extends the capabilities of SDN by enabling the simulation of multiple enclosures connected via apertures. This enhancement allows independent control of the acoustic properties of the individual coupled enclosures and the dimensions of the connecting aperture. The work also derives the transfer function for the SDN system applied to coupled volumes, providing a robust theoretical basis for its functionality.

The effectiveness of the SDN was evaluated by comparing the energy decay curves generated by the system against those obtained from full-scale ray-tracing models and scale model measurements. The results demonstrated that SDN achieves comparable accuracy in simulating complex room acoustics while maintaining significantly lower

computational costs than traditional wave-based or ray-based methods.

This study represents a significant advancement in digital room acoustics modeling by providing a computationally efficient yet accurate method for simulating coupled-volume acoustics. It is particularly relevant for real-time applications where high performance and interactivity are required, such as in virtual environments, architectural acoustic design, and gaming.

2.2.3. Neural Network-Based Methods

Neural network-based methods offer the best trade-off between computational complexity and output quality. In this thesis, the focus is on improving the generation of room impulse responses (RIRs) using Generative Adversarial Networks (GANs), enhancing GAN-generated RIR results, and separating geometric and acoustic modeling through Mesh Transformers.

Ritter et al. [25] introduced a deep neural network capable of learning to dereverberate input audio artificially reverberated with synthetic RIRs. This approach represents one of the earliest applications combining neural networks and RIR processing. Li et al. [26] advanced this area by utilizing GANs specifically to dereverberate audio signals, demonstrating the effectiveness of adversarial learning in handling complex reverberation effects. Meanwhile, Larsson and Viberg [27] proposed a different approach, using two convolutional neural network (CNN)-based autoencoders to directly extract RIRs from sound signals, showcasing the capability of neural architectures to learn detailed room acoustic characteristics. They also developed a process for generating rooms of various shapes. The process creates rooms by generating two rectangles of random but realistic size and either taking the union or the difference of the two rectangles, one of them can be seen in Figure 2.7. This is the first study that generates RIR using neural network.

These works collectively highlight the versatility of neural networks in addressing challenges in room acoustics, from dereverberation to RIR extraction, providing a foundation for the further development of GAN-based RIR generation and the integration of geometric and acoustic modeling techniques.

Papayiannis (2019) [49] employed Generative Adversarial Networks (GANs) to generate artificial room impulse responses (RIRs) with realistic acoustic features. In this approach, the generator uses a normal distribution random input, with no conditional parameters like room geometry or acoustic properties explicitly considered in the generation process.

Luis et al. (2020) explored sound field reconstruction in their study titled "Sound field reconstruction in rooms: inpainting meets super-resolution" [50]. They proposed

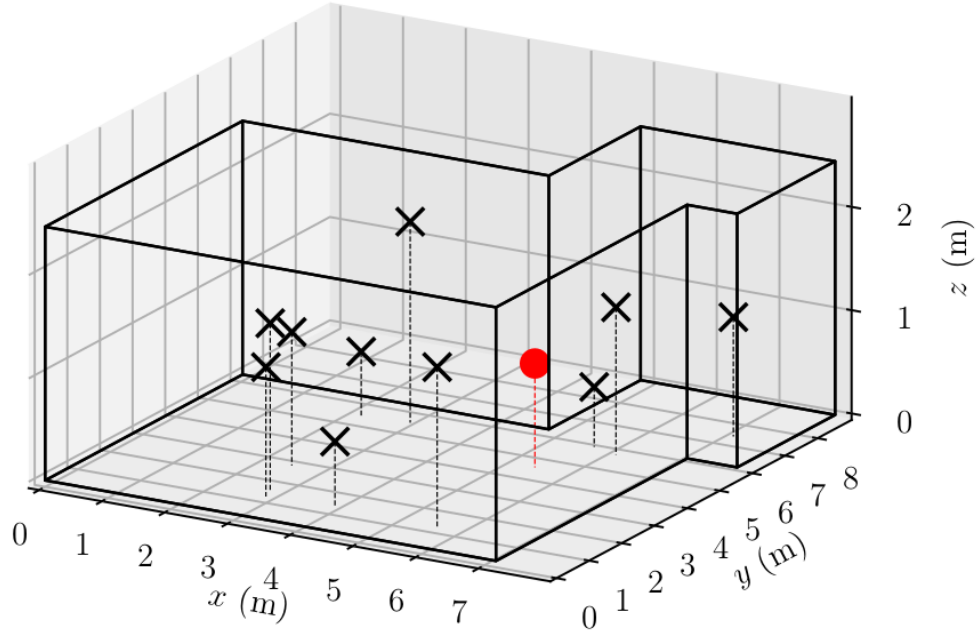


Figure 2.7: Example Generated Room. Red point is the speaker point, crosses are microphone points [27].

a method combining inpainting and super-resolution techniques, leveraging a U-Net architecture to reconstruct sound fields. This work demonstrated how neural networks could be used to enhance spatial sound information.

Wager et al. (2020) proposed a dereverberation method involving the joint estimation of both the dry speech signal and the room impulse response [51]. Their approach integrates deep learning models designed for each task separately and combines them into a joint model with shared parameters, enabling the simultaneous estimation of the clean audio and the acoustic characteristics of the environment. Figure 2.8 illustrates their joint estimation framework, showcasing how it bridges speech and acoustics in a unified learning paradigm.

Singh et al. [28] proposed a novel method to generate Room Impulse Responses (RIRs) from a photograph of the room, as illustrated in Figure 2.9. The approach begins with generating a depth heatmap of the room using an autoencoder. This depth representation serves as input for a conditional GAN, which then produces the corresponding RIR. This method highlights the potential for leveraging visual cues for acoustic simulation.

In parallel, Ratnarajah et al. introduced the IR-GAN [29], a GAN-based framework for generating RIRs. The IR-GAN model is inspired by WaveGAN [52] and uses 100 random input values to generate RIRs that are then compared against real RIRs

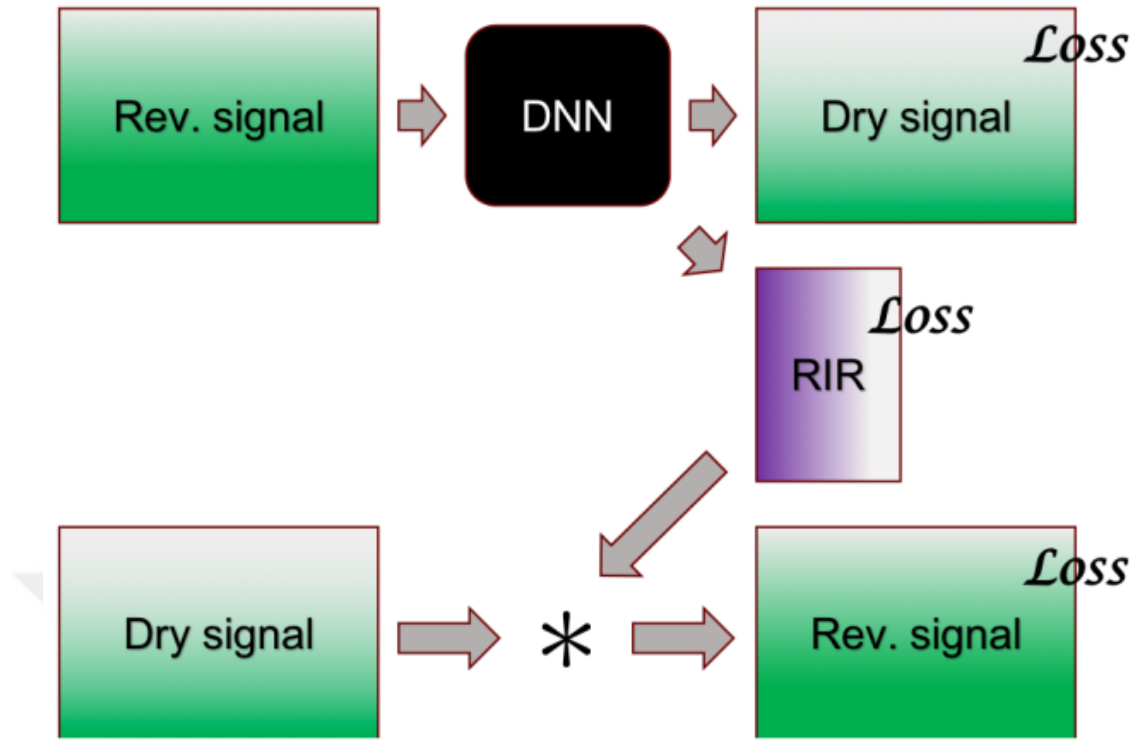


Figure 2.8: Overview of training technique of Wager. The DNN is trained to jointly estimate the dry signal and the RIR. Parameters are shared across the two tasks. The estimated RIR is convolved with the known target dry signal to produce a reverberant signal estimate [51].

from the BUT ReverbDB dataset. These synthetic RIRs were successfully used to lower the Word Error Rate (WER) in Kaldi’s far-field Automatic Speech Recognition (ASR) experiments. Ratnarajah further refined this work by developing TS-RIRGAN [30], which is based on CycleGAN [53] and WaveGAN. This method integrates an encoder, transformer, and decoder (Figure 2.10), achieving significant improvements in the quality of synthetic RIRs. TS-RIRGAN reduced the WER reported by the Kaldi LibriSpeech far-field ASR benchmark by 19

In another study, Ratnarajah et al. introduced FAST-RIR [23], a GAN model derived from the Stage-I component of StackGAN. FAST-RIR (illustrated in Figure 2.11) is the first neural network architecture to generate RIRs by taking room geometry, loudspeaker/microphone coordinates, and RT60 as inputs. Training data consisted of 75,000 synthetic RIRs generated using the Diffuse Acoustic Simulator (DAS) [54], a wave-based RIR generation technique.

RT60, the reverberation time parameter, is central to room acoustics and follows the ISO 3382 standard [55]. It defines the time required for the sound energy in a room to

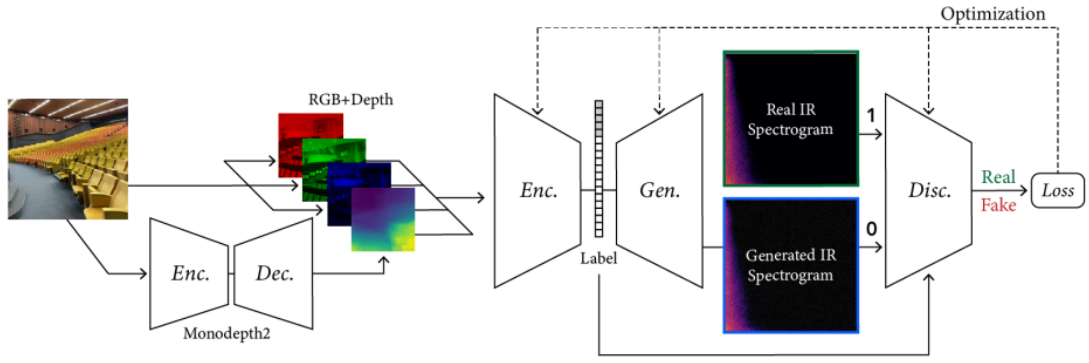


Figure 2.9: Image2Reverb System architecture. The system consists of autoencoder and GAN networks. Left: An input image is converted into 4 channels: red, green, blue and depth. The depth map is estimated by Monodepth2, a pre-trained encoder-decoder network. Right: The model employs a conditional GAN. An image feature encoder is given the RGB and depth images and produces part of the Generator’s latent vector which is then concatenated with noise. The Discriminator applies the image latent vector label at an intermediate stage via concatenation to make a conditional real/fake prediction, calculating loss and optimizing the Encoder, Generator, and Discriminator [28].

decrease by 60 dB after the sound source stops. RT60 values range from fractions of a second to several seconds, depending on the room’s size and construction materials. The ability of FAST-RIR to integrate this parameter as a condition showcases its potential for precise and realistic RIR generation.

The GWA dataset [56] is a comprehensive resource of nearly two million synthetic RIRs, generated from over five thousand room CAD drawings. For each room, RIRs were calculated using various microphone and loudspeaker positions. The dataset employs a hybrid approach for RIR generation: the wave-based finite-difference time-domain (FDTD) method [19] was used for frequencies below 1400 Hz, while a ray-based imaging method was applied for frequencies between 1400 Hz and 8000 Hz. Additionally, acoustic material properties were incorporated into the synthetic RIR generation process, as shown in Figure 2.13. This dataset has been instrumental in advancing neural network-based RIR generation techniques.

Building on the GWA dataset, Ratnarajah introduced the MESH2IR GAN architecture [24], which uses room geometry represented as mesh input rather than simplified

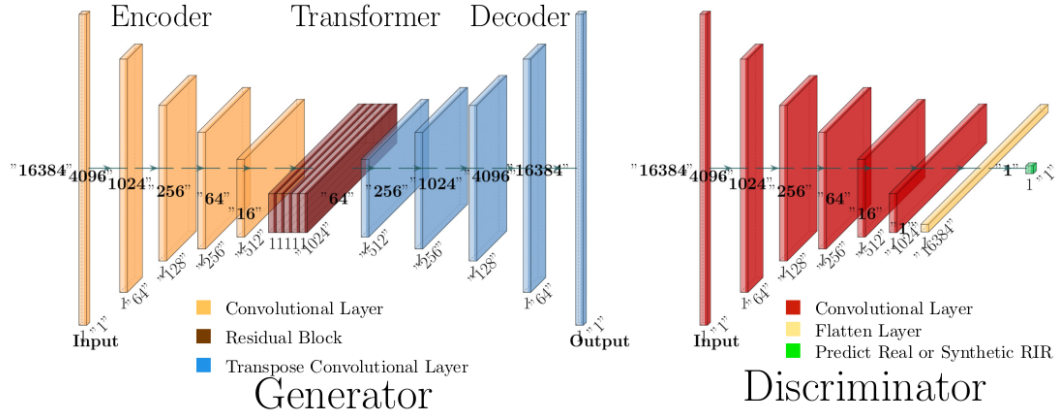


Figure 2.10: The architecture of the generator and discriminator of TS-RIRGAN. The generator takes a synthetic RIR as 1x16384 audio samples and translates it into a real RIR of the same dimension. The discriminator network discriminates between real RIRs and translated synthetic RIRs during training by maximizing the adversarial loss [30].

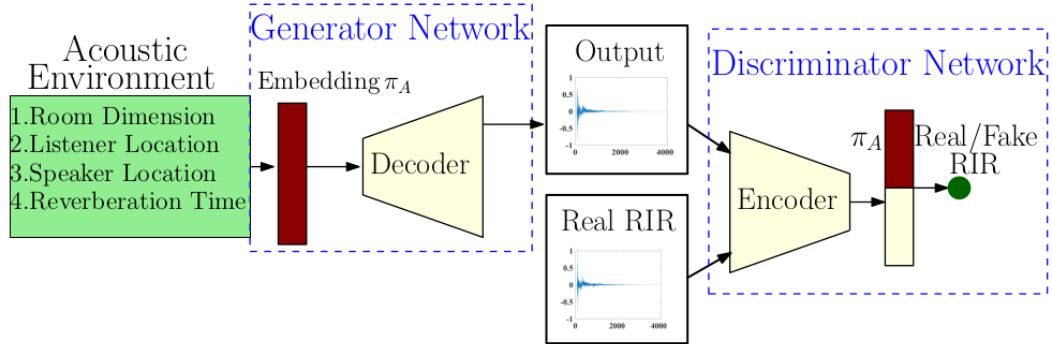


Figure 2.11: The architecture of FAST-RIR model. The Generator network takes acoustic environment details as input and generates corresponding RIR as output. The Discriminator network discriminates between the generated RIR and the ground truth RIR for the given acoustic environment during training [23].

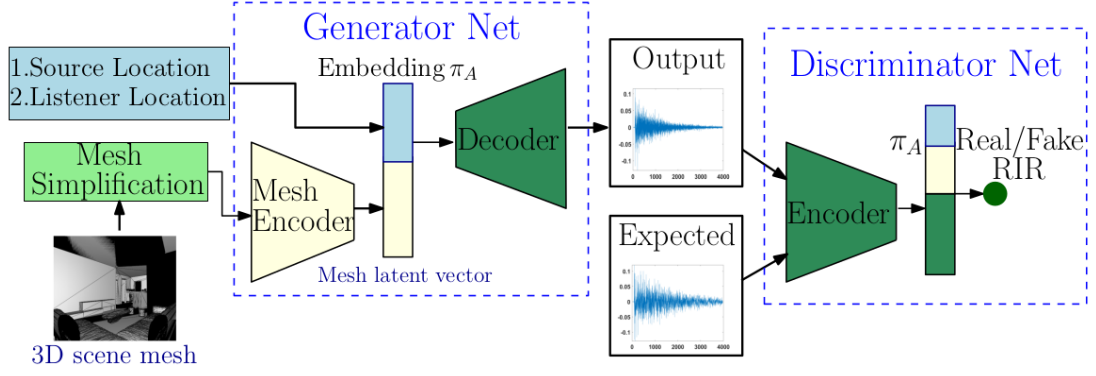


Figure 2.12: The architecture of MESH2IR model. The mesh encoder network encodes a indoor 3D scene mesh to the latent space. The mesh latent vector and the source and listener locations are combined to produce a scene vector embedding (π_A). The generator network generates an IR corresponding to the input scene vector embedding. For the given scene vector embedding, the discriminator network discriminates between the generated IR and the ground truth IR during training [24].

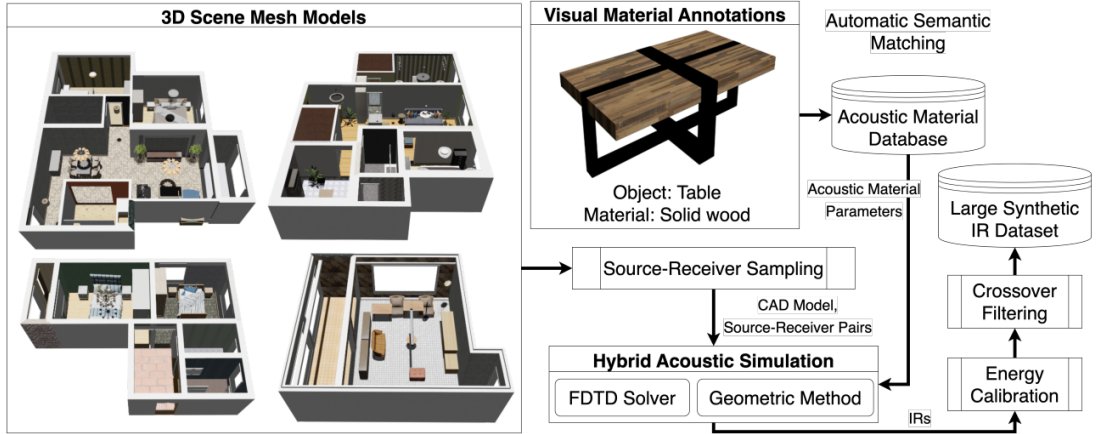


Figure 2.13: GWA model [56] .

parameters such as room dimensions or RT60. This approach captures the intricacies of room geometry more effectively, leading to more accurate RIR generation, as illustrated in Figure 2.12. Notably, MESH2IR is the first GAN-based RIR generator to incorporate a Graph Convolution Network (GCN) [57], which processes the mesh-based room representations. By combining the structural fidelity of GCNs with the generative

capabilities of GANs, MESH2IR achieved significant advancements in RIR modeling. Our work builds on the methodologies of both FAST-RIR and MESH2IR, aiming to refine and extend these approaches.

Further improvements in RIR generation were made by Ratnarajah et al. in their papers "Towards Improved Room Impulse Response Estimation For Speech Recognition" [58] and "Scene2BIR: Material-aware learning-based binaural impulse response generator for reconstructed real-world 3D scenes" [31]. In these studies, energy-based loss functions were introduced to enhance the energy characteristics of generated RIRs. The second paper, Scene2BIR, incorporates detailed acoustic material properties into the neural network architecture, resulting in more realistic RIRs. Although the fundamental GAN structures in these works are similar to those of FAST-RIR and MESH2IR, the inclusion of material-aware features and energy-based loss functions marks a significant step forward in the realism and applicability of synthetic RIRs.

2.2.4. Evaluation of SSIM Loss Function

To the best of our knowledge, FAST-RIR [23] is the first GAN model designed to generate RIRs based on room geometry, microphone position, and loudspeaker position, while MESH2IR [24] is the first GAN model to use a room mesh as input for RIR generation. Both models represent significant advancements in neural network-based RIR synthesis, leveraging the GAN architecture to map complex spatial and acoustic relationships effectively. However, these models employ Mean Squared Error (MSE) as the loss function in their generator networks, which, while suitable for reducing overall prediction error, introduces limitations in the quality of the generated RIRs.

A notable drawback of both FAST-RIR and MESH2IR is the presence of artifacts or glitches in the generated RIRs. These glitches, highlighted in Figure 2.14, undermine the realism and accuracy of the synthetic RIRs, potentially affecting their performance in applications such as speech recognition, auralization, or acoustic analysis. Addressing these glitches is critical to improving the reliability of GAN-based RIR generation techniques.

2.2.4.1. Structural Similarity Index Measure (SSIM)

Mean Squared Error (MSE) is the most widely used metric for assessing image quality, but it does not consider structural differences within an image. Since the human visual system primarily detects structural distortions, Wang et al. introduced the Structural Similarity Index Measure (SSIM) in 2004 as a quality metric for grayscale images, later extended to color images. SSIM analyzes local patterns of pixel intensities, adjusting

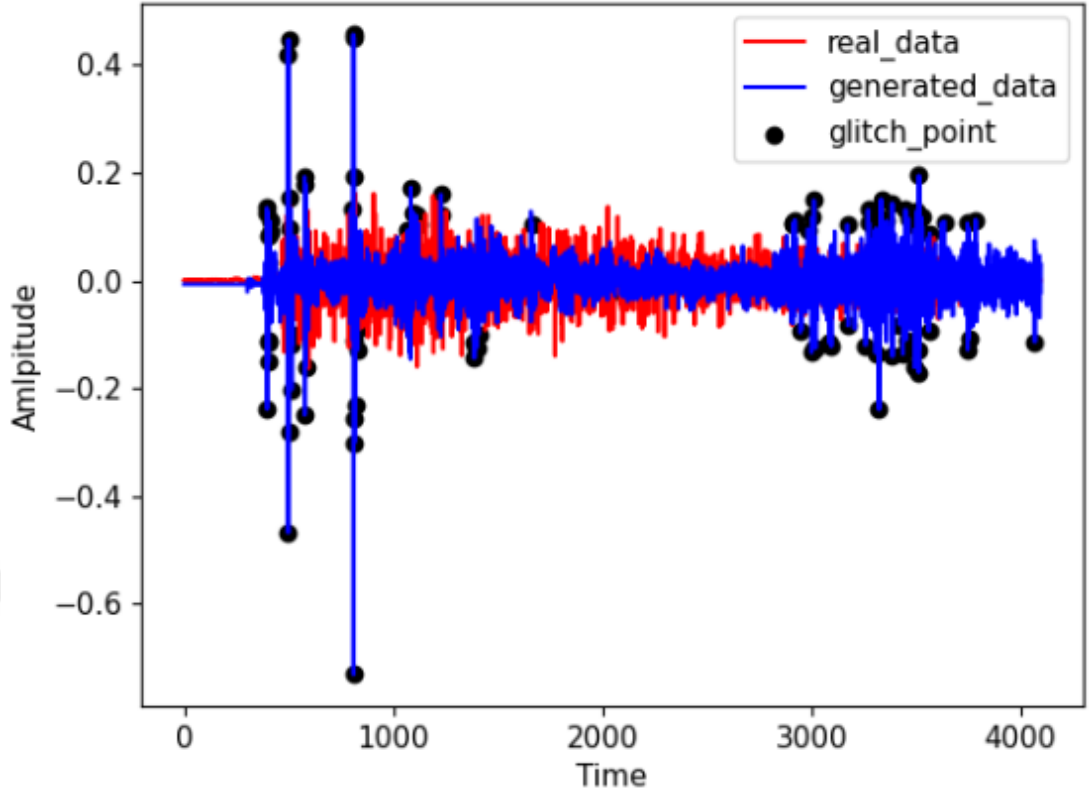


Figure 2.14: Illustration of glitches on a sample RIR data.

for luminance and contrast. Its primary goal is to differentiate between structural and non-structural distortions and measure them separately [61]. SSIM builds upon the Universal Image Quality Index (UQI), which Wang proposed in 2002 [62].

The SSIM metric comprises three key components: luminance, contrast, and structure, as defined in Equation (2.4). Luminance is determined by μ_x (Equation (2.5)) and μ_y , representing the mean intensities of the images, as shown in Equation (2.8). Contrast is based on σ_x (Equation (2.6)) and σ_y , which are the standard deviations—quantifying the variation of pixel intensities from the mean—outlined in Equation (2.9). The structure component depends on σ_{xy} (Equation (2.7)), σ_x , and σ_y , as detailed in Equation (2.10). The structure component is known as the correlation between x and y , with σ_{xy} representing their covariance. Small constants $C1$, $C2$, and $C3$ are introduced to prevent instability in the calculations [61].

$$\begin{aligned}
 SSIM(x, y) &= \left(\frac{2\mu_x\mu_y + C1}{\mu_x^2 + \mu_y^2 + C1} \right) \left(\frac{2\sigma_x\sigma_y + C2}{\sigma_x^2 + \sigma_y^2 + C2} \right) \left(\frac{\sigma_{xy} + C3}{\sigma_x\sigma_y + C3} \right) \\
 &= l(x, y)c(x, y)s(x, y)
 \end{aligned} \tag{2.4}$$

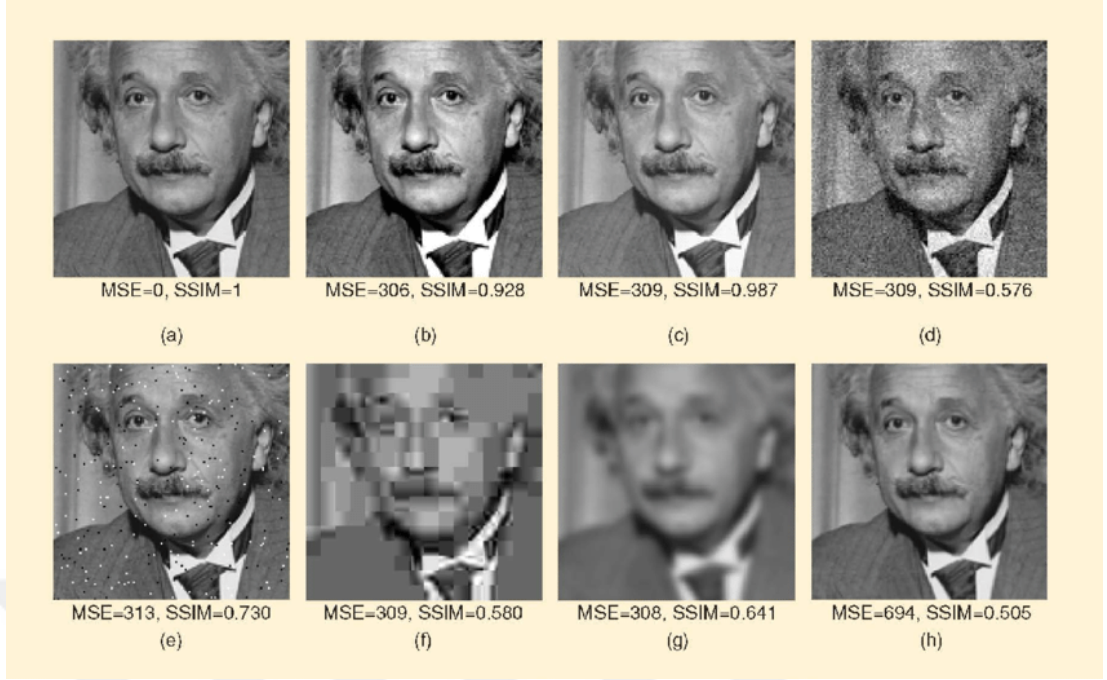


Figure 2.15: Comparison of SSIM and MSE performances for Einstein image altered with different types of distortions. (a) Reference image. (b) Mean contrast stretch. (c) Luminance shift. (d) Gaussian noise. (e) Impulsive noise. (f) JPEG compression. (g) Blurring. (h) Spatial scaling. (Image courtesy of Wang and Bovik's paper [59]) [60].

$$\mu_x = \frac{1}{N} \sum_{i=1}^N x_i \quad (2.5)$$

$$\sigma_x = \left(\frac{1}{N-1} \sum_{i=1}^N x_i - \mu_x \right)^{\frac{1}{2}} \quad (2.6)$$

$$\sigma_{xy} = \frac{1}{N-1} \sum_{i=1}^N x_i - \mu_x \mu_y \quad (2.7)$$

$$l(x, y) = \left(\frac{2\mu_x\mu_y + C1}{\mu_x^2 + \mu_y^2 + C1} \right) \quad (2.8)$$

$$c(x, y) = \left(\frac{2\sigma_x\sigma_y + C2}{\sigma_x^2 + \sigma_y^2 + C2} \right) \quad (2.9)$$

$$s(x, y) = \left(\frac{\sigma_{xy} + C3}{\sigma_x\sigma_y + C3} \right) \quad (2.10)$$

Cosine similarity quantifies the cosine of the angle between two vectors, computed as the dot product of the vectors divided by the product of their magnitudes. It is formally defined in Equation (2.11) and can be associated with correlation as $s(x, y) = correlation(x, y) = CosineSimilarity(x - \mu_x, y - \mu_y)$ [63]. This suggests that the structure component $s(x)$ can also be computed using the cosine similarity metric [64].

$$CosineSimilarity(x, y) = \frac{\sum x_i y_i}{\sqrt{\sum x_i^2} \sqrt{\sum y_i^2}} \quad (2.11)$$

Luminance measures the similarity in brightness between two images, while structure assesses the consistency of structural information by analyzing pixel intensity patterns. Contrast quantifies the similarity in intensity variations. The SSIM value ranges from -1 to 1, where 1 represents perfect similarity, 0 indicates no correlation, and -1 signifies complete dissimilarity. The luminance (l) and contrast components range from [0,1], whereas the structure (s) component ranges from [-1,1]. Unlike MSE, SSIM is more resilient to variations in brightness and contrast as it prioritizes local patterns that align with human visual perception. However, despite its effectiveness in evaluating perceptual quality, SSIM does not fully capture the diversity or distributional properties of the generated data [65].

SSIM highlights structural variations by incorporating standard deviation, which quantifies the dispersion of pixel intensities from the mean, averaging over mean values, or calculating the correlation between images. These characteristics make SSIM robust against non-structural changes that have minimal impact on human perception, such as contrast stretching, or the addition of a constant value to all pixels [61].

Mathematically, Brunet and Vrscay have proven that SSIM represents a partial ordering, satisfying the following conditions, which make it a potential candidate for use as a loss function [66]:

1. $SSIM(x, y) = 0 \Leftrightarrow x = y$
2. $SSIM(x, y) = SSIM(y, x)$
3. $SSIM(x, y) + SSIM(y, z) \succcurlyeq SSIM(x, z)$

SSIM metric in Equation (2.4) is applied on a single patch of two images. Total SSIM value of two images is obtained by convolving a gaussian filter with window size 11x11 and stride 1, then by applying SSIM formula of luminance, contrast and structure as shown in Equation (2.4), on each window and averaging over the number of windows (M) as shown in Equation (2.12). This is may be called on some papers as Mean SSIM (MSSIM) [61]. The pytorch implementation of ssim, which we utilize in our paper [67],

follows this formula [68].

$$MSSIM(x, y) = \frac{1}{M} \sum_{j=1}^M SSIM(x_j, y_j) \quad (2.12)$$

For image quality assessment, applying the SSIM index locally on image patches rather than globally is more effective. This approach is beneficial for several reasons. First, image statistical features are often highly spatially nonstationary. Second, image distortions, whether dependent or independent of local image statistics, can vary across different regions. Third, at typical viewing distances, the human visual system can only perceive a limited area of the image with high resolution at any given moment. Lastly, localized quality measurement enables the creation of a spatially varying quality map, which provides more detailed insights into image degradation and can be valuable for various applications [61].

On the other hand, several studies have identified limitations of the SSIM metric. Jean-François Pambrun and Rita Noumeir, in their paper "Limitations of the SSIM Quality Metric in the Context of Diagnostic Imaging", provided both analytical and empirical evidence highlighting the following shortcomings of SSIM [69]:

1. The luminance term can exhibit instability or insensitivity in areas where the average intensity is either very low or very high.
2. The variance term can be unstable in areas with low variance.
3. The variance and structure terms tend to underestimate distortions near sharp edges.

Due to these limitations, SSIM by itself is not suitable for replacing MSE in rate-distortion optimization algorithms used in image coding. It is recommended to use SSIM alongside another metric, such as MSE, to provide a more thorough evaluation of image quality [69].

Nilsson and Akenine-Möller, in their paper Understanding SSIM, examined the mathematical properties of SSIM and showed that it can yield unexpected, undefined, or nonintuitive results in both synthetic and real-world scenarios. In the third and fifth rows of Figure 2.16, they found that when the entire image has low luminance (close to zero), SSIM produces very low values, even if the images appear visually similar. This issue stems from the quadratic normalization denominator $(\mu_x^2 + \mu_y^2 + C1)$ in Equation (2.4), which exaggerates differences in dark regions. As a result, SSIM does not always match the human visual system's (HVS) perception of luminance differences [70].

There are several variations of SSIM designed for different application scenarios in image similarity. These include Multiscale SSIM (MS-SSIM) [71], Complex Wavelet

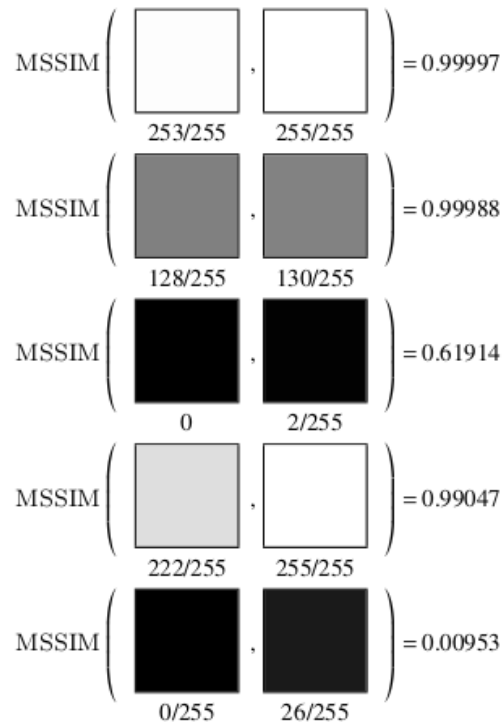


Figure 2.16: MSSIM behavior for constant grayscale images. The grayscale value is reported below each image. Note that in the third row, it is difficult to see any differences between the images, and still, MSSIM values indicate low quality. In the fourth row, it is clearly easy to see a difference between the images, while MSSIM indicate that they are similar. At the bottom, we see that when comparing a black image to an image with 26/255, MSSIM is almost zero [70].

SSIM (CW-SSIM) [72], 3D-SSIM [73], Spherical SSIM [74], Micro SSIM [75], Color SSIM (CMSSIM) [76], Quaternion SSIM [77], HSSIM [78], and Channel-Wise SSIM [78], among others. There are also studies aimed at improving the SSIM metric itself [79–81]. In the paper A Hitchhiker’s Guide to Structural Similarity, Venkataramanan et al. discussed and compared these different SSIM variations [78]. In this thesis, we apply the SSIM metric to audio data.

2.2.4.2. Audio Similarity Measurement using SSIM

All the models discussed earlier utilize Mean Squared Error (MSE) as the loss function in their generator networks. In our work, we explore the use of the Structural Similarity Index Measure (SSIM) as an alternative loss function within the generative component of the GAN. SSIM, which is traditionally applied to 2D images, aligns closely with human perceptual judgments, as illustrated in Figure 2.15. To adapt SSIM for 1D audio data, there are two main strategies. The first involves reshaping the 1D audio data from a shape of (N) to a 2D form of $(N,1)$. The second approach transforms the audio data into the frequency domain, using techniques such as Short-Time Fourier Transform (STFT), Frequency Spectrogram, or Mel-Frequency Cepstral Coefficients (MFCC).

Breakey and Meskel [82] found that applying SSIM to the frequency domain, rather than directly to time-domain signals, provided greater robustness against random noise, frequency shifts, and time lags. Kandadi et al. [83] supported these findings, noting that applying SSIM variations to spectrograms produced similar results. Gan et al. [84] further demonstrated that applying SSIM to the real part of the FFT was highly correlated with subjective audio quality assessments and outperformed temporal SSIM. Additionally, Zhu et al. [85] introduced a new audio quality assessment algorithm, Structural Similarity Analysis of Modulation (SSAM), which extends the analysis of structural similarity across three dimensions: time, frequency, and modulation frequency domains. Sheng et al. [86] combined MSE and SSIM loss functions in a CGAN model after transforming audio signals into Mel spectrograms. Their method outperformed WaveNet vocoders in subjective Mean Opinion Score (MOS) evaluations of synthesized speech.

Guided by these findings, we conducted experiments to evaluate the incorporation of SSIM into the loss function for generating Room Impulse Responses (RIRs). Our experiments explored two approaches: (1) augmenting the 1D RIR data to a 2D format with shape $(N,1)$ as mentioned in [87], and (2) extracting MFCC features from the audio data. SSIM was then applied to these representations to assess its influence on both the perceptual and quantitative quality of the generated RIRs.

MFCCs, commonly used in audio analysis, capture the mean frequency cepstral coef-

ficients over time intervals. The MFCC spectrogram of a signal, as shown in Figure 2.17, provides a compact yet comprehensive frequency representation, making it a suitable candidate for SSIM-based loss evaluation. This approach aims to improve the quality of GAN-generated RIRs by leveraging SSIM’s alignment with human perceptual judgment.

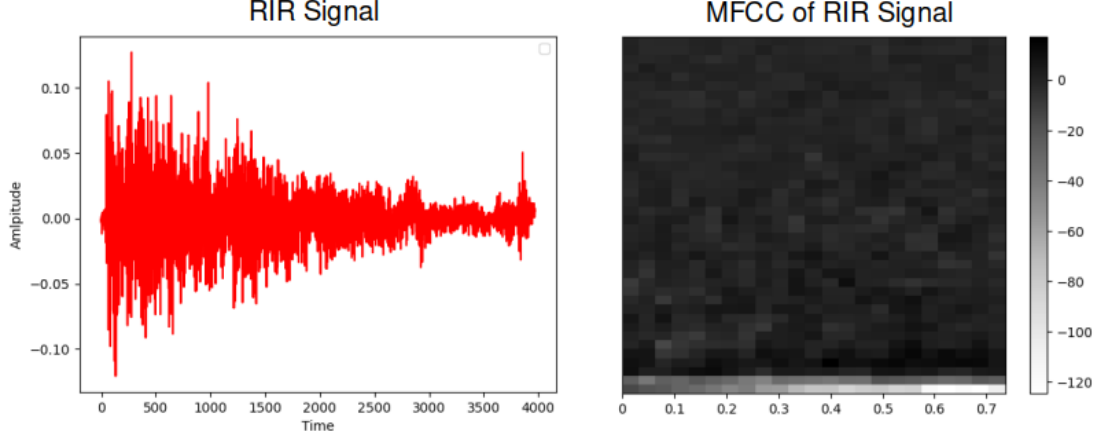


Figure 2.17: MFCC of a sample RIR with 40 coefficients.

The use of Structural Similarity Index Measure (SSIM) as a loss function in neural networks is not entirely new, as shown by previous research. Zhao et al. [88] explored SSIM loss functions in multi-color image restoration, demonstrating that combining Mean Squared Error (MSE) and SSIM as a loss function outperforms training with either MSE or SSIM alone. Similarly, Nilsson and Möller [89] analyzed the behavior of the SSIM similarity metric and concluded that SSIM by itself does not provide significant advantages over MSE in multi-color image applications.

Our study aligns with these findings, demonstrating that a hybrid loss function combining both MSE and SSIM yields better results than using either function alone. Additionally, this work is the first to explore SSIM as a loss component in the generation of Room Impulse Responses (RIRs). This novel approach showcases SSIM’s potential to enhance both the perceptual and quantitative quality of generated RIRs, drawing on the strengths of existing research in the image processing field.

2.2.4.3. Strengths and Weaknesses of SSIM in Audio Data

In subsection 2.2.4.1, we discussed the advantages and limitations of the SSIM metric in image analysis. In this section, we will examine its strengths and weaknesses in assessing audio similarity. Additionally, we will compute MSE along with the individual

luminance, contrast, and structure components of SSIM. Covariance values will also be provided to better identify the source of errors within each component.

We selected a reference RIR signal (Figure 2.18) from the GTU-RIR dataset and applied the following artificial distortion methods to analyze the behavior of MSE, SSIM, and its individual components—Luminance, Contrast, Structure, and Covariance—under these conditions:

1. Mean contrast stretch: The amplitude of the RIR signal was amplified by a factor of 1.89, as shown in Figure 2.19 (a).
2. Mean contrast stretch applied to low-amplitude regions: The amplitude of the last half of the RIR signal (low-amplitude region) was amplified by a factor of 3, as shown in Figure 2.19 (b).
3. Mean contrast stretch applied to high-amplitude regions: The amplitude of the first half of the RIR signal (high-amplitude region) was amplified by a factor of 1.98, as shown in Figure 2.19 (c).
4. Luminance shift: A constant value of 0.063 was added to the entire RIR signal, as shown in Figure 2.20 (a).
5. Luminance shift applied to low-amplitude regions: A constant value of 0.727 was added to the last half of the RIR signal (low-amplitude region), as shown in Figure 2.20 (b).
6. Luminance shift applied to high-amplitude regions: A constant value of 0.127 was added to the first half of the RIR signal (high-amplitude region), as shown in Figure 2.20 (c).
7. Gaussian noise: A noise array of the same length as the RIR signal was generated using a Gaussian normal distribution with mean 0 and standard deviation 0.0635, then added to the signal, as shown in Figure 2.21 (a).
8. MP3 compression: The RIR signal was compressed using the MP3 algorithm then decompressed, as shown in Figure 2.21 (b).
9. Blurring: A Gaussian filter with a standard deviation of 3.6 was convoluted with the RIR signal via convolution, as shown in Figure 2.21 (c).

All distortions were adjusted so that they resulted in an MSE value of 0.0040, allowing for a fair comparison of the SSIM metric. The results are summarized in Table 2.3. As shown in Table 2.3, the Luminance term in the SSIM metric is the most dominant component.

Table 2.3: MSE and SSIM values for various distortion types.

Distortion Type	MSE	SSIM	Luminance	Contrast x Structure	Covariance	Glitch Count
Reference	0	1	1	1	0.0035	0
Mean contrast stretch	0.0040	0.8720	0.9237	0.9425	0.0066	116
Mean contrast stretch applied to low-amplitude	0.0040	0.8110	0.8811	0.9115	0.0049	130
Mean contrast stretch applied to high-amplitude	0.0040	0.9204	0.9548	0.9602	0.0062	145
Luminance shift	0.0040	0.3390	0.3390	1.0000	0.0035	0
Luminance shift applied to low-amplitude	0.0040	0.4222	0.4223	0.9999	0.0035	0
Luminance shift applied to high-amplitude	0.0040	0.7758	0.7760	0.9999	0.0035	0
Gaussian noise	0.0040	0.6931	0.8053	0.8571	0.0034	33
MP3 compression	0.0040	0.7701	0.8528	0.8861	0.0013	58
Blurring	0.0040	0.7889	0.8949	0.8659	0.0001	95

When mean contrast stretching is applied to the entire RIR signal, as shown in Figure 2.19 (a), the SSIM value remains close to 1, as expected. However, when the same amount of change is applied only to the second half of the signal, where smaller values exist

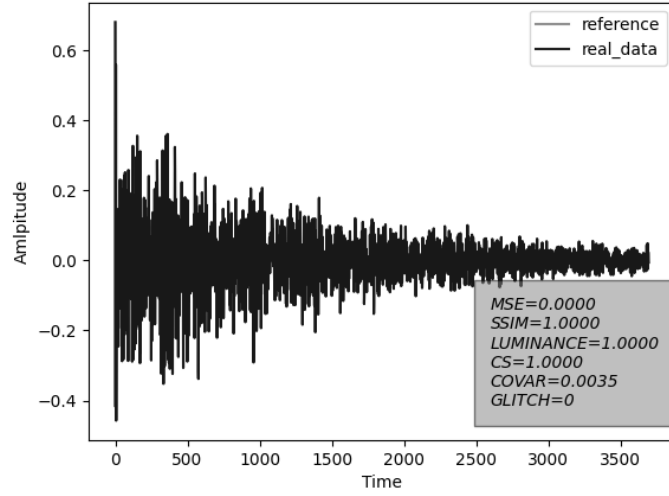


Figure 2.18: Reference RIR signal.

(Figure 2.19 (b)), the SSIM value decreases, effectively detecting the change. This demonstrates SSIM's strength when used as a loss function, as it helps the neural network recognize such variations.

Conversely, when the same modification is applied to the first half of the RIR signal, where large amplitudes exist (Figure 2.19 (c)), the SSIM value increases, making it ineffective for detecting the anomaly. This highlights one of SSIM's weaknesses when used as a loss function. The reason for this limitation is that SSIM primarily captures local proportional changes. For high-amplitude regions, local differences may not be significant enough to be detected by SSIM, as it is applied locally and normalized by $(\mu_x^2 + \mu_y^2 + C1)$ and $\sigma_x^2 + \sigma_y^2 + C2$.

When we apply a mean luminance shift to the entire RIR signal, as shown in Figure 2.20 (a), the SSIM value is low, unlike in image processing. This difference arises because, in images, the mean intensity is less perceptually significant, whereas in audio, the mean amplitude plays a crucial role in perception, making the shifted signal sound different from the reference.

When the same shift is applied only to the second half of the signal, where smaller values exist (Figure 2.20(b)), the SSIM value remains low, effectively detecting the change. However, when the shift is applied to the first half of the RIR signal, where larger amplitudes dominate (Figure 2.20(c)), the SSIM value remains high. This prevents the neural network from effectively detecting the anomaly when SSIM is used as a loss function, highlighting another weakness of the SSIM metric.

When we apply Gaussian noise (Figure 2.21 (a)), MP3 compression (Figure 2.21(b))

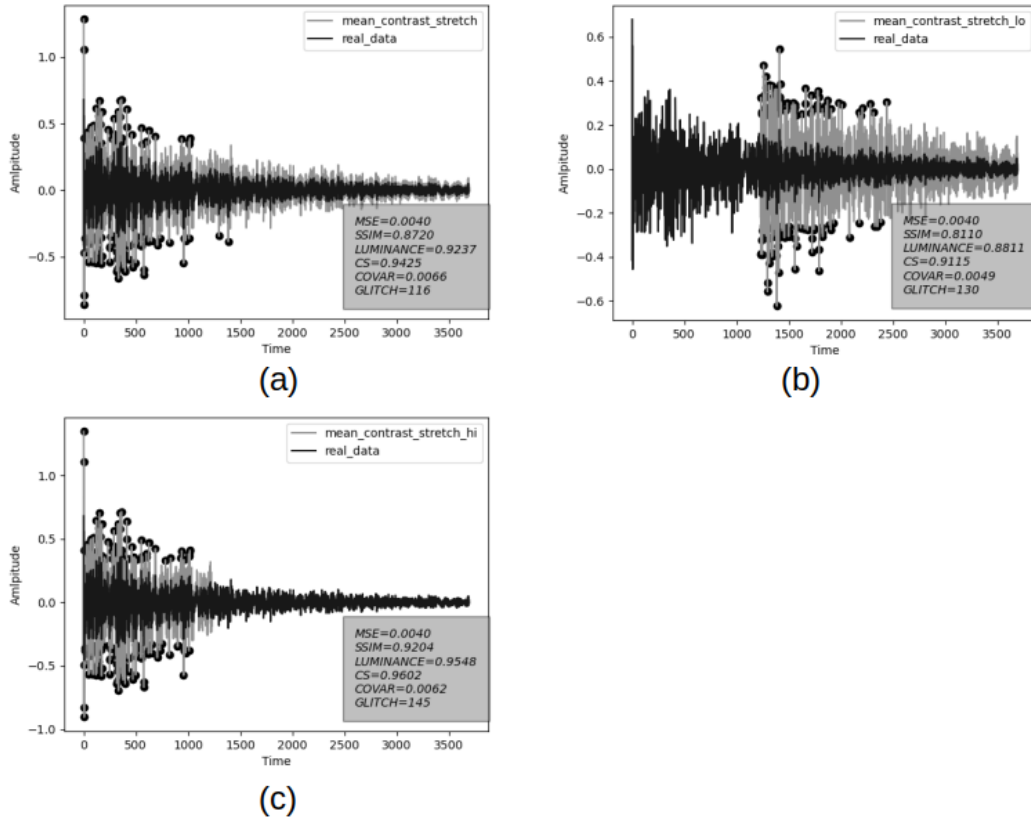


Figure 2.19: Distortions of the Mean Contrast Stretch type: (a) Mean contrast stretch, (b) Mean contrast stretch applied to low-amplitude regions, and (c) Mean contrast stretch applied to high-amplitude regions. For the signal in (c), SSIM values remain very high, indicating that this metric is not sufficiently effective in detecting the anomaly.

and blurring (Figure 2.21(c)), the SSIM metric behaves similarly to how it does in the image domain, providing more meaningful results.

2.2.5. Seperating Geometric and Acoustic Models

Using the GWA dataset [56] as training data, Ratnarajah introduced the MESH2IR GAN architecture [24], which utilizes room geometry in the form of mesh input. This work marked the first use of a Graph Convolution Network (GCN) [57] in a GAN framework for RIR generation. Neural network-based RIR generators that leverage room mesh geometry include "MESH2IR: Neural Acoustic Impulse Response Generator for Complex 3D Scenes" [24], "Towards Improved Room Impulse Response Estimation

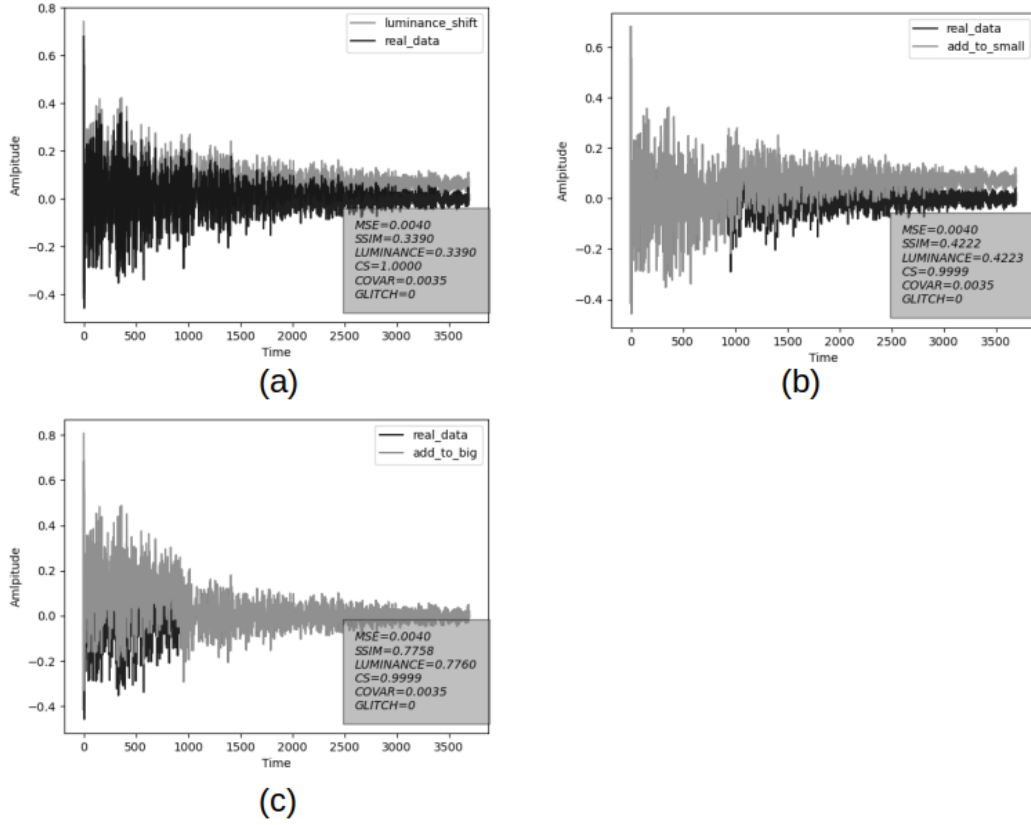


Figure 2.20: Distortions of the Luminance Shift type: (a) Luminance shift applied to the entire signal. (b) Luminance shift applied to the low-amplitude region. (c) Luminance shift applied to the high-amplitude region. For the signal in (c), the SSIM values remain very high, indicating that this metric is not effective enough to detect the anomaly.

For Speech Recognition” [58], and ”Listen2Scene: Interactive material-aware binaural sound propagation for reconstructed 3D scenes” [31]. The Listen2Scene paper [31] further incorporates acoustic material properties into the neural network to enhance RIR realism. However, the foundational architecture of these models remains consistent with the MESH2IR design.

In a more recent advancement, Ratnarajah et al. extended this approach in the paper ”AV-RIR: Audio-visual room impulse response estimation” [32], integrating visual data captured from the scene to enhance RIR generation, an idea inspired by Singh et al. [28]. Singh’s model generated RIR from a room photo by first producing a depth heatmap using an autoencoder and subsequently employing a conditional GAN for RIR synthesis. A common characteristic of these neural network models is the combined training of

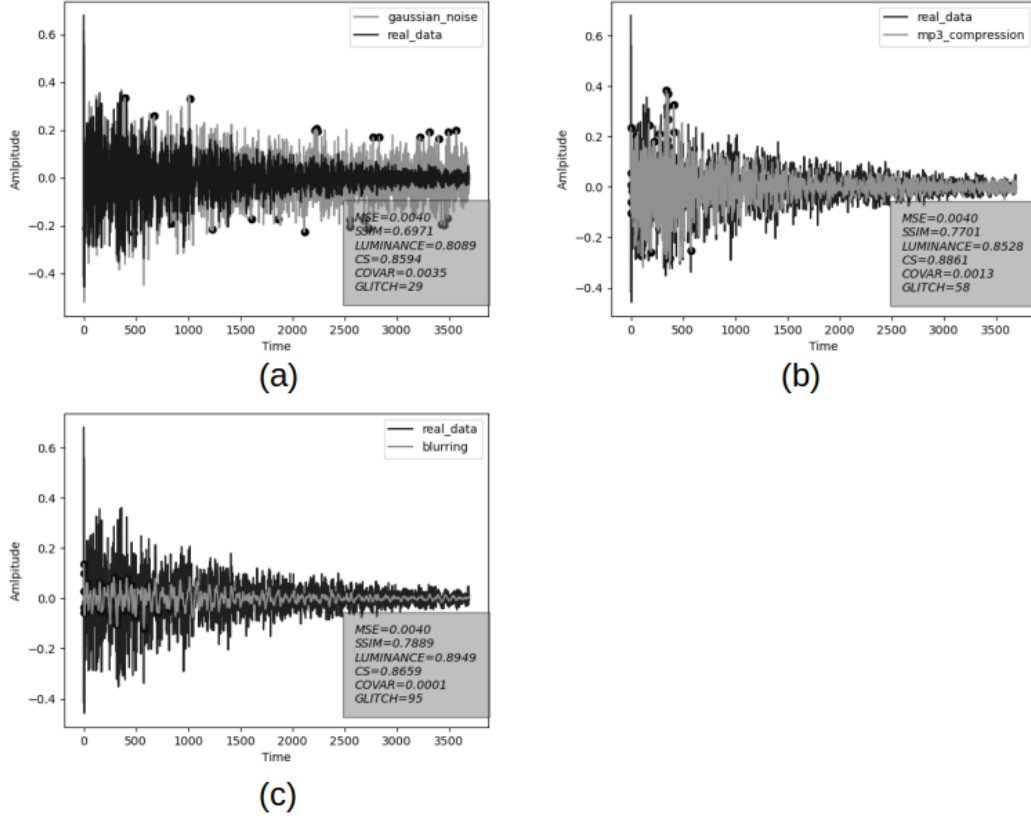


Figure 2.21: Various types of distortions: (a) Gaussian Noise. (b) MP3 Compression. (c) Blurring.

the geometric (e.g., GCN) and acoustic (e.g., StackGAN) components, necessitating the re-training of the entire network for different room geometries.

To address the limitations of tightly coupled geometric and acoustic models, we explored autoencoder frameworks for geometric data based on graph neural networks, transformers, and neural radiance fields (NERF) [90]. Existing graph neural network-based autoencoders presented challenges: they either failed to simultaneously reconstruct node features and adjacency matrices or supported only limited reconstruction cases [91–101]. Given these constraints, we investigated both mesh-based and graph-based methods, as their compatibility offers a promising pathway for decoupling the geometric and acoustic models, as illustrated in Figure 2.22.

In their work, Kipf and Welling introduced the Variational Graph Auto-Encoders (VGAE) model, which is specifically designed to reconstruct only the adjacency matrix A , rather than the node feature matrix X that represents the coordinates or other attributes of the nodes in the graph. The model aims to learn a probabilistic representation of the graph’s structure, but it does not directly address the reconstruction of the node feature

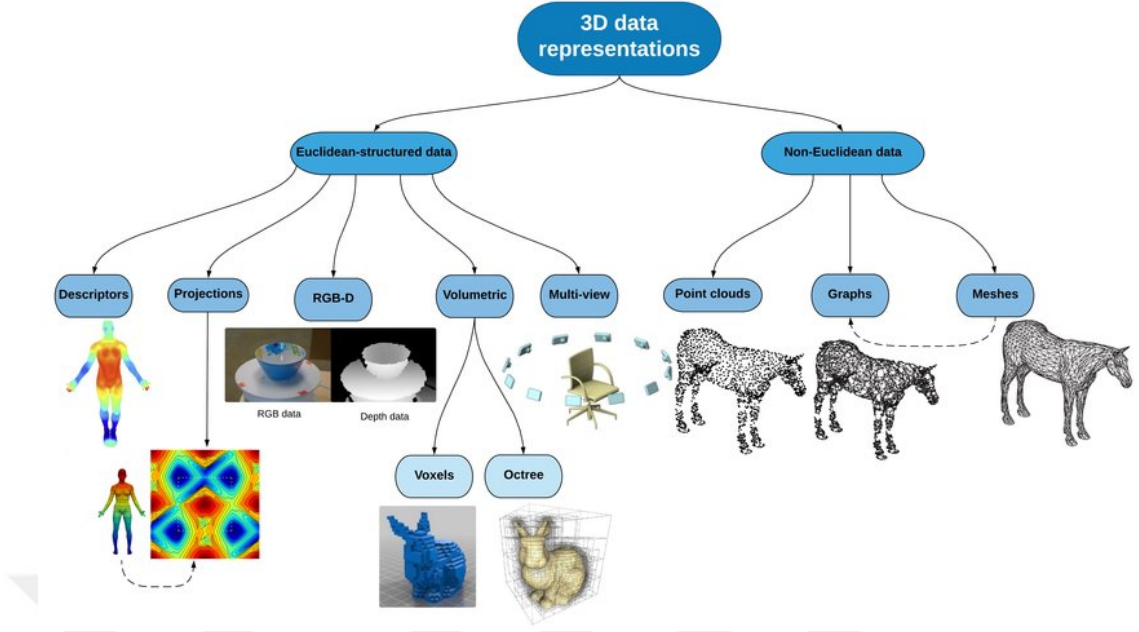


Figure 2.22: Various representations for 3D data: Euclidean representation (Descriptors, Projections, RGB-D, Volumetric; voxels and octree and multi-view) and Non-Euclidean representations (Point clouds, graphs and meshes) [102].

matrix, which would include geometric information in the case of room geometry. A non probabilistic version of their model is shown in equations 2.14 and 2.13 [91]. In this equation , D is degree, A is adjacency, X is per-node feature, W is weight matrix.

This formulation, however, focuses solely on learning a graph’s structure through the adjacency matrix A , without including node-specific features X , which are crucial for representing geometric data in the context of room acoustics. Therefore, this limitation restricts the model’s ability to effectively learn and reconstruct full geometric data, which would include the positioning of nodes or elements (such as the coordinates of room surfaces).

$$Z = GCN(X, A) = D^{-1/2}(A + I)D^{-1/2}XW \quad (2.13)$$

$$\tilde{A} = \sigma(ZZ^T) \quad (2.14)$$

Zhou et al., in their review paper ”Graph Neural Networks: A Review of Methods and Applications” [93], outlined five methods for graph generation, but none focused on

simultaneously reconstructing both the adjacency matrix A (representing connectivity) and the feature matrix X (representing node-level attributes such as geometry or other properties). This limitation prevents these methods from being fully applicable to applications requiring a complete graph reconstruction that includes both structure and attributes.

Wu et al., in their comprehensive review of graph autoencoders (GAEs) [94], summarized 13 different graph autoencoder architectures. However, like the methods discussed by Zhou et al., these models focus on reconstructing either A or X independently and do not provide a unified framework for reconstructing both.

Litany et al., in their work on deformable mesh autoencoders [95], focused on reconstructing mesh geometries but only partially rebuilt the input mesh, making their approach unsuitable for tasks requiring full recovery of both connectivity and feature matrices.

Simonovsky and Komodakis, in their "GraphVAE" framework [96], used variational autoencoders to generate small graphs, reconstructing only the adjacency matrix A while neglecting the feature matrix X .

Salha et al., in their work "Keep It Simple: Graph Autoencoders Without Graph Convolutional Networks" [97], demonstrated the reconstruction of adjacency matrix A without employing graph convolutional networks (GCNs). However, they also did not address the reconstruction of feature matrix X .

Zhang et al., in their 2020 paper "Graph Deconvolutional Networks" [103], defined graph deconvolution in terms of reconstructing the feature matrix ($\tilde{X}$) from the latent representation Z produced by the GCN encoder as shown in equation 2.15.

Similarly, Salha et al., in their later work "Simple and Effective Graph Autoencoders with One-Hop Linear Models" [100], proposed several GAE variants that focus only on adjacency matrix reconstruction, without tackling the recovery of the feature matrix X .

In summary, existing graph generation and autoencoder methods focus on reconstructing either AA or XX but not both simultaneously. This limitation poses a significant challenge for applications like RIR generation that require comprehensive geometric and acoustic representations.

$$\tilde{X} = D^{-1/2}(A + I)D^{-1/2}ZW_1 \quad (2.15)$$

Li et al., in their paper "Deconvolutional Networks on Graph Data" [99], proposed a novel framework for graph neural networks that incorporates deconvolutional operations, allowing for both the restoration of adjacency matrix A and feature matrix X . This work

aimed to address the limitation of prior graph autoencoders by enabling the simultaneous reconstruction of graph connectivity and node features.

In their subsequent paper "Graph Autoencoders with Deconvolutional Networks" [98], Li et al. extended this approach by offering a complete framework for graph autoencoder architectures. This framework includes both an encoder and decoder. Encoder part of this framework is given in the equations 2.16, 2.17, and 2.18. Decoder part is given in the equations 2.19 (unpooling) and 2.20. Loss function is given in the equation 2.21. Despite the promise of this framework, in our experiments, we failed to reconstruct the mesh of apartment flats accurately.

$$H = GCN(X, A) \quad (2.16)$$

$$S = \text{pooling} = \text{softmax}(\tanh(HW_1)W_2) \quad (2.17)$$

$$Z = S^T H; A^{\text{pool}} = S^T A S \quad (2.18)$$

$$H' = SZ; A' = S A^{\text{pool}} S^T \quad (2.19)$$

$$X' = GDN(A', H') = (I + \text{laplacian_normalization}(A'))H'W_3 \quad (2.20)$$

$$\text{Loss} = \lambda_A \text{MSE}(A, A') + \lambda_X \text{MSE}(X, X') \quad (2.21)$$

Studies that combine graph neural networks (GNNs) and transformers have shown promise in various applications, but they fall short when tasked with reconstructing full graphs, including both node properties and edges.

Yun et al., in their paper "Graph Transformer Networks" [104], focused on using GCNs and transformers to generate adjacency matrices. However, their approach did not extend to reconstructing node feature matrices, limiting its utility for tasks requiring complete graph reconstruction.

Mialon et al. provided insights into integrating transformers with graph data in their work "GraphIT" [105]. While they did not explicitly operate on 3D mesh data, they demonstrated key findings:

- Vanilla transformers equipped with appropriate positional encoding and structural information outperform GNNs in tasks such as graph classification and regression.
- Kernel-based relative positional encoding proved more effective than absolute positional encoding, such as those derived from Laplacian eigenvectors.
- Combining structural encoding schemes within node features, like graph convolutional kernel networks (GCKN), with relative positional encoding yielded the best results for encoding graph structure in transformers.

Lin et al. proposed a novel pipeline that leverages transformer architectures to process images and generate 3D meshes. In their study [106], the input was an image, and the output was a 3D human mesh (Figure 2.23). They further refined the accuracy of their results by embedding GCN modules within the transformer architecture (Figure 2.24), demonstrating the potential synergy between these models for tasks involving geometric reconstructions.

Despite these advancements, existing methods integrating GNNs and transformers have yet to achieve complete graph reconstructions with both node properties and edge connectivity in a unified framework. This underscores the need for further exploration, particularly in adapting these models to handle complex 3D mesh data and diverse graph representations.

Zhang et al., in their paper "Graph Masked Autoencoders with Transformers" [107], focused on reconstructing the feature vector X of masked nodes in a graph. Their approach benefits from reduced memory consumption through masking, making it computationally efficient. Khajenezhad et al. [108] introduced a graph generation model based on a single transformer encoder, designed to reconstruct missing parts of graphs. Meanwhile, Rampasek et al. proposed a powerful and scalable graph transformer architecture called GPS, which achieves state-of-the-art results on diverse benchmarks while maintaining linear complexity [109]. Similar approaches leveraging structural encodings and transformers have been explored in [110–113].

Older methods like CNN-based models that operate on mesh data, such as those proposed in [114, 115], lack the capability to fully reconstruct mesh data. Similarly, NERF-based models for mesh processing [90, 116, 117] were not applicable to our problem, as they focus primarily on neural rendering and volumetric representation rather than mesh reconstruction.

Among newer approaches, mesh transformers such as those described in "MeshMAE: Masked Autoencoders for 3D Mesh Data Analysis" [118] and "MeshGPT: Generative Pre-training for 3D Mesh Data" [119], provide a more relevant framework for our task. Specifically, the ideas from MeshMAE were directly applicable to our problem of mesh-based reconstruction.

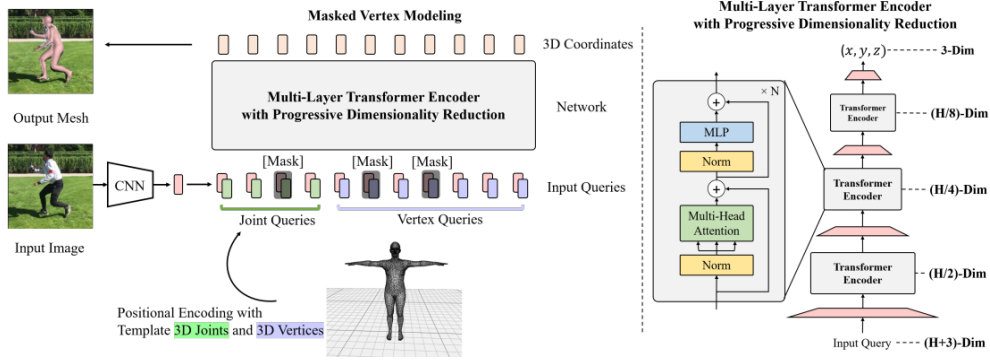


Figure 2.23: Overview of the proposed framework of Lin et al. [106]. Given an input image, they extract an image feature vector using a convolutional neural network (CNN). They perform position encoding by adding a template human mesh to the image feature vector by concatenating the image feature with the 3D coordinates (x_i, y_i, z_i) of every body joint i , and 3D coordinates (x_j, y_j, z_j) of every vertex j . Given a set of joint queries and vertex queries, we perform self-attentions through multiple layers of a transformer encoder, and regress the 3D coordinates of body joints and mesh vertices in parallel. They use a progressive dimensionality reduction architecture (right) to gradually reduce the hidden embedding dimensions from layer to layer. Each token in the final layer outputs 3D coordinates of a joint or mesh vertex. Each encoder block has 4 layers and 4 attention heads. H denotes the dimension of an image feature vector.

Building on the MESH2IR and MeshMAE frameworks, we replaced the GCN component of the MESH2IR network with a mesh transformer auto-encoder inspired by MeshMAE. This new network, which we named MESH-TAE, combines the strengths of mesh transformers with the acoustic modeling capabilities of GANs, enabling more accurate and efficient reconstruction of 3D mesh data in the context of room impulse response generation.

2.3. Summary

This chapter provides a comprehensive review of the existing literature on room impulse response (RIR) measurement and generation techniques. It begins with an overview of fundamental concepts, including the definitions of reverberation, RIR, and reverberation time (RT60), laying the groundwork for understanding the importance of RIR in acoustic

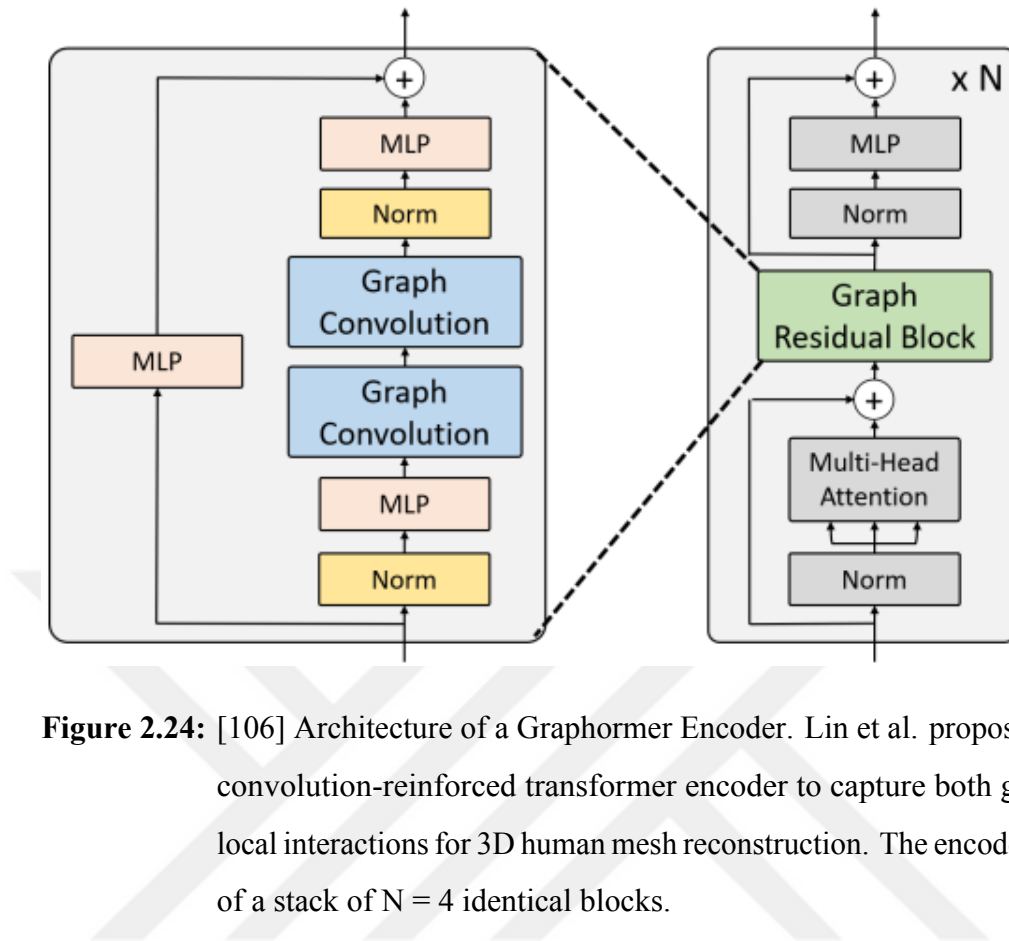


Figure 2.24: [106] Architecture of a Graphormer Encoder. Lin et al. propose a graph-convolution-reinforced transformer encoder to capture both global and local interactions for 3D human mesh reconstruction. The encoder consists of a stack of $N = 4$ identical blocks.

analysis and simulation.

The chapter then delves into various methodologies and source signals used in RIR measurement, including maximum length sequence (MLS), time delay spectrometry (TDS), and exponential sine sweep (ESS). These methods are discussed in terms of their principles, applications, and limitations.

In the section on RIR generation methods, traditional approaches such as ray-based and wave-based techniques are explored alongside state-of-the-art neural network-based methods. Particular attention is given to the application of the Structural Similarity Index Metric (SSIM) for mitigating glitches in generated RIRs. The discussion highlights the challenges in ensuring the perceptual and acoustic fidelity of the generated signals.

Furthermore, the chapter reviews geometric embedding techniques for modeling room geometry, focusing on two primary approaches: graph neural network-based models and mesh transformer-based architectures. The strengths and limitations of these methods are analyzed, with the chapter concluding that mesh transformer-based architectures provide a more robust solution for RIR generation due to their enhanced ability to represent complex 3D room geometries.

In summary, this chapter encapsulates the state of research on RIR measurement and generation, offering insights into methodologies, challenges, and advancements. It underscores the potential of mesh transformer-based models to address current limitations and pave the way for improved acoustic simulation and RIR generation.



3. GTU-RIR DATASET

Publicly available RIR datasets are summarized in Table 3.1. These datasets serve as valuable resources for benchmarking and testing RIR generation methods. However, to address certain limitations in existing datasets and to provide a more versatile testing platform, we developed a new dataset called GTU-RIR (Gebze Technical University Room Impulse Response).

The GTU-RIR dataset is designed to test RIR generation algorithms against real-world data while offering the flexibility to be reanalyzed from different perspectives. It comprises a larger number of data points compared to currently available datasets and demonstrates improved spatial distribution. This enhanced spatial diversity allows for a more comprehensive evaluation of RIR generation methods.

A comparative analysis of the spatial coordinate distributions between the BUT ReverbDB dataset and the GTU-RIR dataset is illustrated in Figures 3.1 and 3.2. These visualizations highlight the broader and more uniform spatial coverage achieved by the GTU-RIR dataset, underscoring its potential as a robust benchmark for future research in RIR generation.

Table 3.1: Public RIR Datasets [120] [121]

DATASET	Number of RIRs	LICENSE
360° Binaural Room Impulse Response (BRIR) Database for 6DOF spatial perception research	1726	CC BY 4.0
Aachen Impulse Response Database	214	Unknown
Audio Spatialisation for Headphones (ASH) Impulse Response Dataset	787	CC BY-CC-SA 4.0
BUT Speech@FIT Reverb Database	2325	CC BY 4.0
Concert Hall Impulse Responses – Pori, Finland	90	Custom, similar to CC BY-NC-SA
DRR-scaled Individual Binaural Room Impulse Responses	4936	CC BY-NC-SA 4.0
Database of Omnidirectional and B-Format Impulse Responses	2041	Unknown
Dataset of In-The-Ear and Behind-The-Ear Binaural Room Impulse Responses	192	CC BY-NC 4.0

Table 3.1: (continued) Public RIR Datasets [120] [121]

DATASET	Number of RIRs	LICENSE
Dataset of measured binaural room impulse responses for use in an position-dynamic auditory augmented reality application	3888	CC BY-NC 4.0
EchoThief Impulse Response Library	115	Unknown
Greg Hopkins IR 1 – Digital, Analog, Real Spaces	22	Unknown
GTU-RIR Gebze Technical University Room Impulse Reponse Dataset	15000	GPL
Impulse Response Database for HybridReverb2	472	CC BY-SA 4.0
Impulse Responses from the Bell Labs Varechoic Chamber	12	Unknown
METU SPARG Eigenmike em32 Acoustic Impulse Response Dataset v0.1.0	8052	CC BY 4.0
Multi-Channel Impulse Response Database	1872	Unknown
Multichannel Acoustic Reverberation Database at York	72	Unknown
Open Acoustic Impulse Response (Open AIR) Library	504	Unknown
R-Prox RIR samples Darmstadt June 2017	2313	CC BY 4.0
REVERB challenge RealData	36	Unknown
RIR samples Darmstadt and Helsinki, Summer-Autumn 2018	1788	CC BY 4.0
RWCP Sound Scene Database in Real Acoustical Environments	6758	Unknown
Single- and Multichannel Audio Recordings Database (SMARD)	1008	Unknown
Statistics of natural reverberation enable perceptual separation of sound and space	270	Unknown
Surrey Binaural Room Impulse Response Measurements	370	MIT
The IoSR listening room multichannel BRIR dataset	3456	CC BY 4.0

Table 3.1: (continued) Public RIR Datasets [120] [121]

DATASET	Number of RIRs	LICENSE
Voxengo Free Reverb Impulse Responses	38	Custom, similar to CC BY-SA

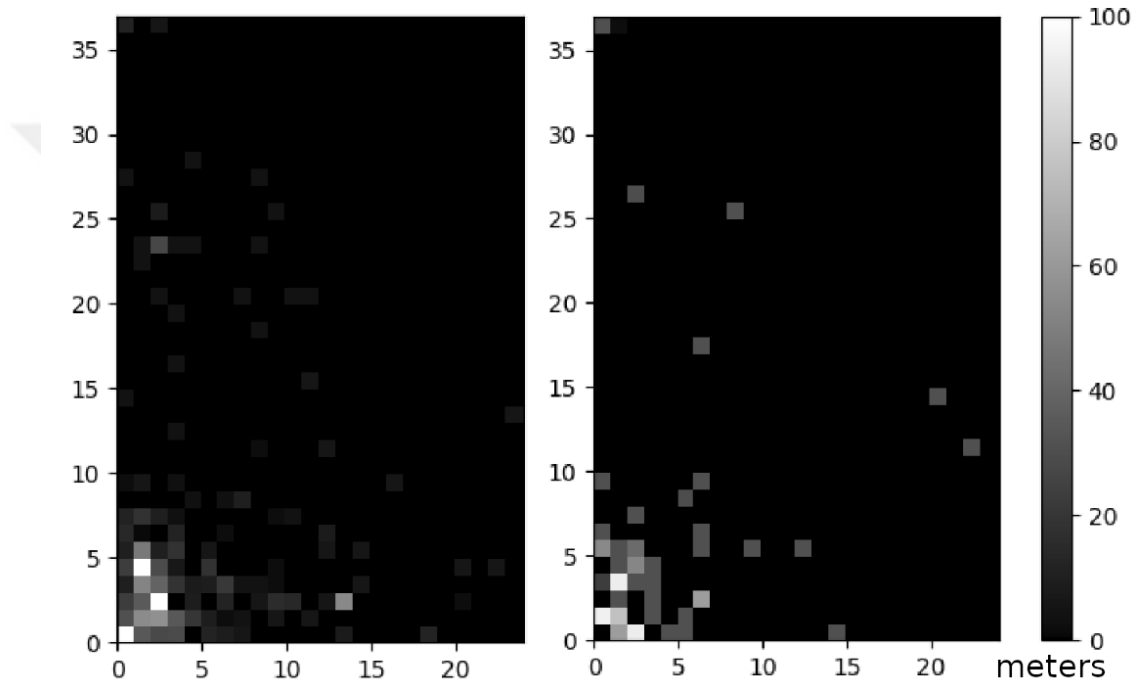


Figure 3.1: Heatmap displaying the spatial distribution of microphones and loudspeakers in the BUT ReverbDB data. (left: microphone, right: loudspeaker). Average distance between microphone and loudspeaker is 4.4 meters.

3.1. Room Impulse Response Measurement

Room impulse responses (RIRs) capture the acoustic footprint of a room, encapsulating how sound propagates from a source to a receiver within that space. These responses are crucial for several applications, including improving room acoustics, generating reverberated sound specific to the room, or removing reverberation from recorded audio.

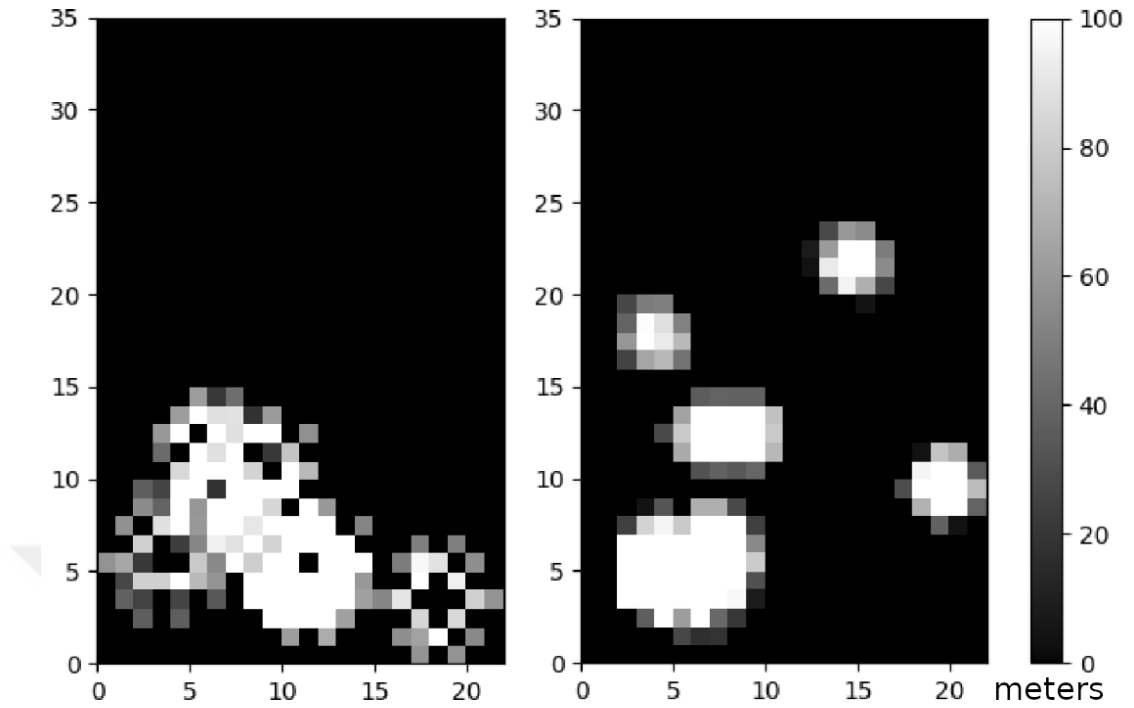


Figure 3.2: Heatmap displaying the spatial distribution of microphones and loudspeakers in the GTU-RIR data. (left: microphone, right: loudspeaker). Average distance between microphone and loudspeaker is 4.6 meters.

An RIR is specific to a unique combination of parameters: the room geometry, surface reflection coefficients, the loudspeaker location, and the microphone location. To compute the entire sound field in a room, RIRs need to be measured for various combinations of these parameters. Since these parameters are continuous by nature, their values must be discretized for practical measurement. The choice of interval between discrete values can be adjusted based on computational resources, with smaller intervals yielding more detailed data at the expense of computational effort.

The reason for using an impulse signal, as opposed to an ordinary sound signal, lies in achieving a high signal-to-noise ratio (SNR). Additionally, measurements are often conducted during the quietest times to minimize background noise, ensuring accurate RIR recordings.

During RIR measurement, the room, the source, and the receiver are considered a unified system, as illustrated in Figure 3.3. The system's response to an impulse provides a comprehensive characterization of the acoustic properties of the room, which can then be analyzed or used in subsequent applications [37].

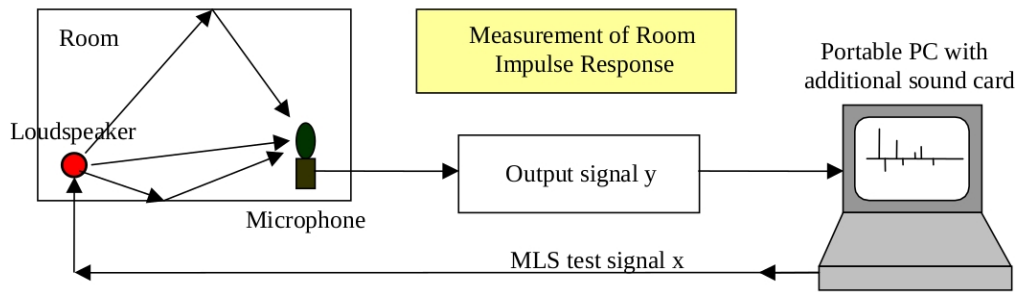


Figure 3.3: RIR measurement of a system of room geometry and source and microphone locations [122].

In practical scenarios, generating a perfect impulse for testing purposes is not feasible. Instead, a brief pulse is often employed as an approximation of an ideal impulse. As long as this pulse is sufficiently short relative to the system's impulse response duration, the resulting measurement will closely resemble the theoretical impulse response.

However, there are challenges with using a very short and intense pulse, particularly in systems susceptible to nonlinear behavior. Such strong pulses can push the system into a nonlinear operating regime, compromising the accuracy of the measurement. To address this, pseudo-random sequences are frequently used as input signals. These sequences, like Maximum Length Sequences (MLS) or Exponential Sine Sweeps (ESS), provide a more controlled way to stimulate the system. The impulse response is then derived mathematically by analyzing the relationship between the input pseudo-random sequence and the system's output [123].

3.1.1. Theoretical Explanation of RIR Measurement

The reverberation characteristics of a closed environment, such as a small room, theatre, or concert hall, can be analyzed under the assumption that the room's geometry and the objects within it remain unchanged. For each specific position of the sound transmitter (loudspeaker) and receiver (microphone), a static system is defined along with the room. The only time-dependent variables are environmental noise and room temperature. However, we assume that environmental noise is negligible compared to the transmitted sound and that the room temperature remains stable through air conditioning systems. Under these conditions, the system is considered time-invariant.

Additionally, the system can be classified as linear because the reverberated signal is the superposition (summation) of the direct and reflected signals within the room [124]. Therefore, this system is referred to as a Linear Time-Invariant (LTI) system

[125]. We also assume that the transmitter and receiver remain stationary. To ensure these assumptions hold, we conducted our experiments at night when environmental noise and temperature variations were minimal, and the transmitter/receiver positions remained unchanged while conducting each step of the experiment (while recording the reverberation for a specific position of loudspeaker/microphone).

As shown in Figure 3.4, an LTI system is represented by H , where x is the input signal and $y = H(x)$ is the corresponding output signal [125]. The output y is also referred to as the system's response to the input x . Here, x represents the sound signal transmitted from the loudspeaker, while y is the reverberated sound signal received by the microphone. In practice, we send a signal x from the speaker into the system and record the resulting output signal y until the reverberation fully decays. A time-invariant system that takes in signal $x(t)$ and produces output $y(t)$ will also, when excited by signal $x(t + \sigma)$, produce the time-shifted output $y(t + \sigma)$ which means that if we the transmitted sound is delayed for a time T , then the system will give us exactly the same reverberated signal with time shifting of T . [125].

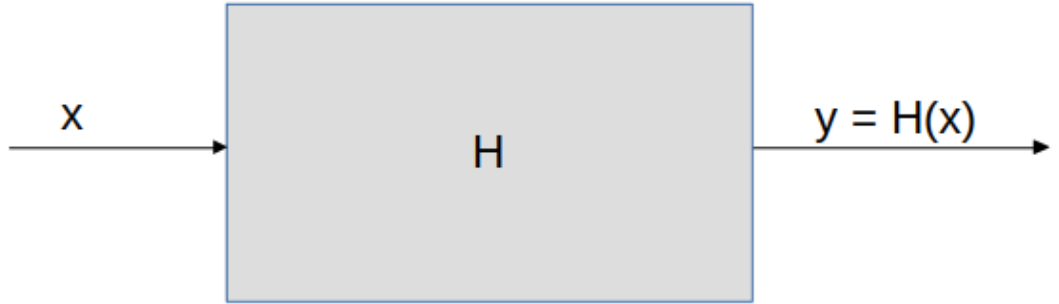


Figure 3.4: Linear Time Invariant System (H) [125].

Linear time-invariant (LTI) systems have the algebraic properties expressed in Equations (3.1) to (3.3). In these equations T denotes the time shift for Continuous-Time systems, and r denotes time shift for Discrete-Time systems. When the input signals are continuous then the system is called as a Continuous-Time LTI system, and when the input signals are discrete then the system is called as a Discrete-Time LTI system [125].

1. Linear :

$$H(\alpha x_1 + \beta x_2) = \alpha H(x_1) + \beta H(x_2) \quad (3.1)$$

2. Time-invariant :

(a) Continuous-Time :

$$y(t) = H(x(t)) \Rightarrow \forall T, y(t - T) = H(x(t - T)) \quad (3.2)$$

(b) Discrete-Time :

$$y[n] = H(x[n]) \Rightarrow \forall r, y[n - r] = H(x[n - r]) \quad (3.3)$$

The system encompassing the room geometry, speaker, and microphone is not a continuous system but a discrete one because both the transmitter (loudspeaker) and the receiver (microphone) operate using discrete digital signals. Therefore, we model the system as a Discrete-Time Linear Time-Invariant (LTI) system as shown in Figure 3.6. A discrete signal $x[n]$ is the discrete representation of a continuous signal x as shown in Figure 3.5.

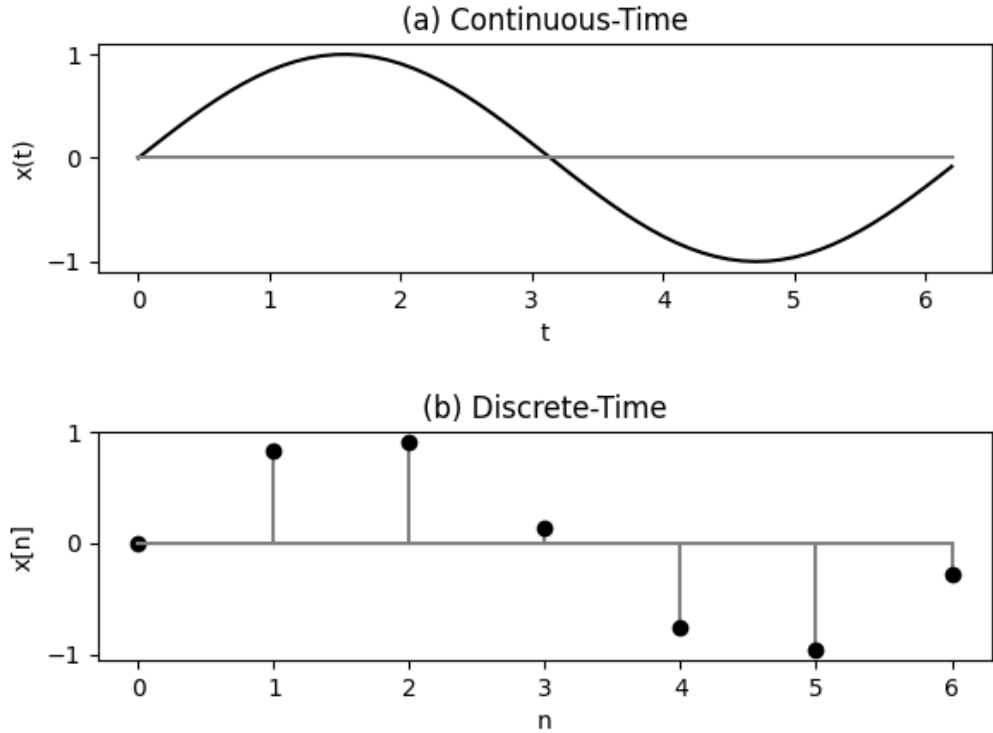


Figure 3.5: Continuous-Time and Discrete-Time signals [125]

Any discrete signal can be considered as a sequence of individual unit impulses, as illustrated in Figure 3.5. These discrete impulses can be represented using the Kronecker delta function [126]. The Kronecker Delta is formally defined in Equation (3.4). While it is traditionally expressed using the indices i, j , it can also be formulated using $n, 0$,

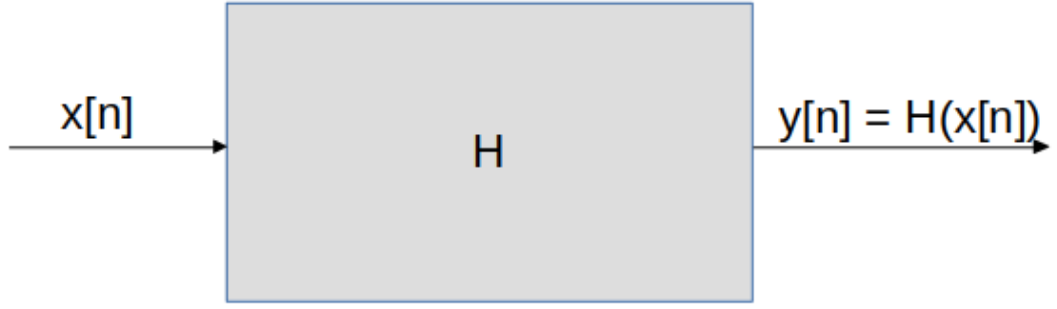


Figure 3.6: Discrete-Time Linear Time Invariant System [125]

where n denotes the index at which the function takes the value of 1. Additionally, the index notation can be written using square brackets instead of subscripts, as shown in Equation (3.4). It is important to note that the Dirac Delta function is used in the sampling of continuous signals [125].

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases} \quad (3.4)$$

Using the definition of unit impulse (Kronecker Delta) we can consider the input signal $x[n]$ as the sum of the unit impulses $\delta[n - k]$ and amplitudes at that point $x[k]$ as shown in Equation (3.5). This summation is illustrated in Figure 3.7 which corresponds to the representation of an arbitrary sequence as a linear combination of shifted unit impulses $\delta[n - k]$, where the weights in this linear combination are $x[k]$ [125].

$$x[n] = \sum_{k=-\infty}^{\infty} x[k] \delta[n - k] \quad (3.5)$$

The output $y[n]$ of a Discrete-Time LTI system H can be formulated as shown in Equations (3.6) to (3.11) [125]. The transition from Equation (3.7) to Equation (3.8) is due to the linearity property of the LTI system. The transition from Equation (3.8) to Equation (3.9) occurs because the system H depends only on n and not on k . The transition from Equation (3.9) to Equation (3.10) is based on the fact that $h_k[n]$ represents the response of the linear system to the shifted unit impulse $\delta[n - k]$. Finally, the transition from Equation (3.10) to Equation (3.11) follows from the time-invariance property of the system, meaning that since $\delta[n - k]$ is a time-shifted version of $\delta[n]$, the response $h_k[n]$ is also a time-shifted version of $h[n]$ [125].

$$y[n] = H(x[n]) \quad (3.6)$$

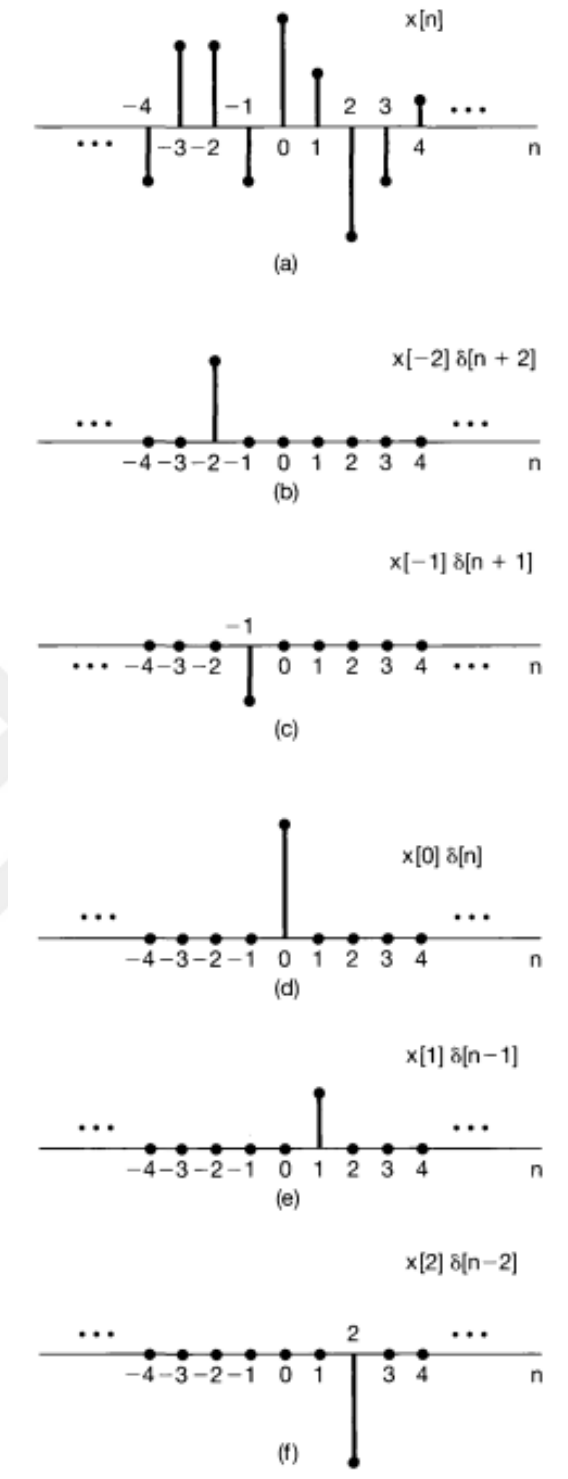


Figure 3.7: Decomposition of a discrete-time signal into a weighted sum of shifted impulses [125].

$$y[n] = H\left(\sum_{k=-\infty}^{\infty} x[k]\delta[n-k]\right) \quad (3.7)$$

$$y[n] = \sum_{k=-\infty}^{\infty} H(x[k]\delta[n-k]) \quad (3.8)$$

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]H(\delta[n-k]) \quad (3.9)$$

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h_k[n] \quad (3.10)$$

$$y[n] = \sum_{k=-\infty}^{\infty} x[k]h[n-k] \quad (3.11)$$

This result is known as the convolution sum or superposition sum, and the operation on the right-hand side of Equation (3.11) is referred to as the convolution of the sequences $x[n]$ and $h[n]$. The convolution operation is symbolically represented as shown in Equation (3.12) [125].

$$y[n] = x[n] * h[n] \quad (3.12)$$

Why is measuring the Impulse Response sufficient to determine the behavior of an LTI system? The Equations (3.6) to (3.11) are summarized in Figure 3.8. In Figure 3.9, the left side illustrates the input signal represented as a series of impulses. Each individual impulse generates a response that is delayed by the same amount as the corresponding input impulse, as shown on the right. The total output is then the sum of these individual responses, forming the convolution sum [125]. The “impulse response” of a system, $h[n]$, is defined as the output produced in response to an impulse input, $x[n] = \delta[n]$. For a unit impulse input $\delta[n]$, which is nonzero only when $n = 0$, the output is $y[n] = h[n]$. Consequently, to determine the impulse response of a system, you simply feed it with $x[n] = \delta[n]$. This principle underpins the use of very short sound events to measure impulse responses.

When computing a convolution sum, the input signal is essentially decomposed into a sequence of states, and the system’s response to each state is determined by its impulse response. Therefore, knowing the response to a single impulse is sufficient to characterize the entire system, as expressed in Equation (3.13), where $h[n]$ represents the impulse response [125].

A fundamental result in LTI system theory states that any LTI system can be completely described by a single function, known as the system’s impulse response. The system output $y[n]$ is simply the convolution of the input $x[n]$ with the impulse response $h[n]$ [127].

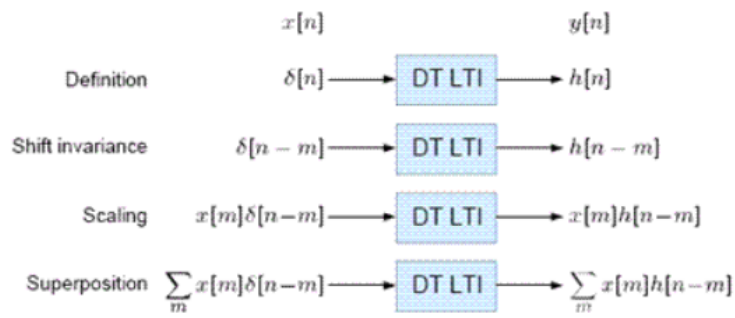


Figure 3.8: Discrete-Time Linear Time-Invariant system principles summary [125].

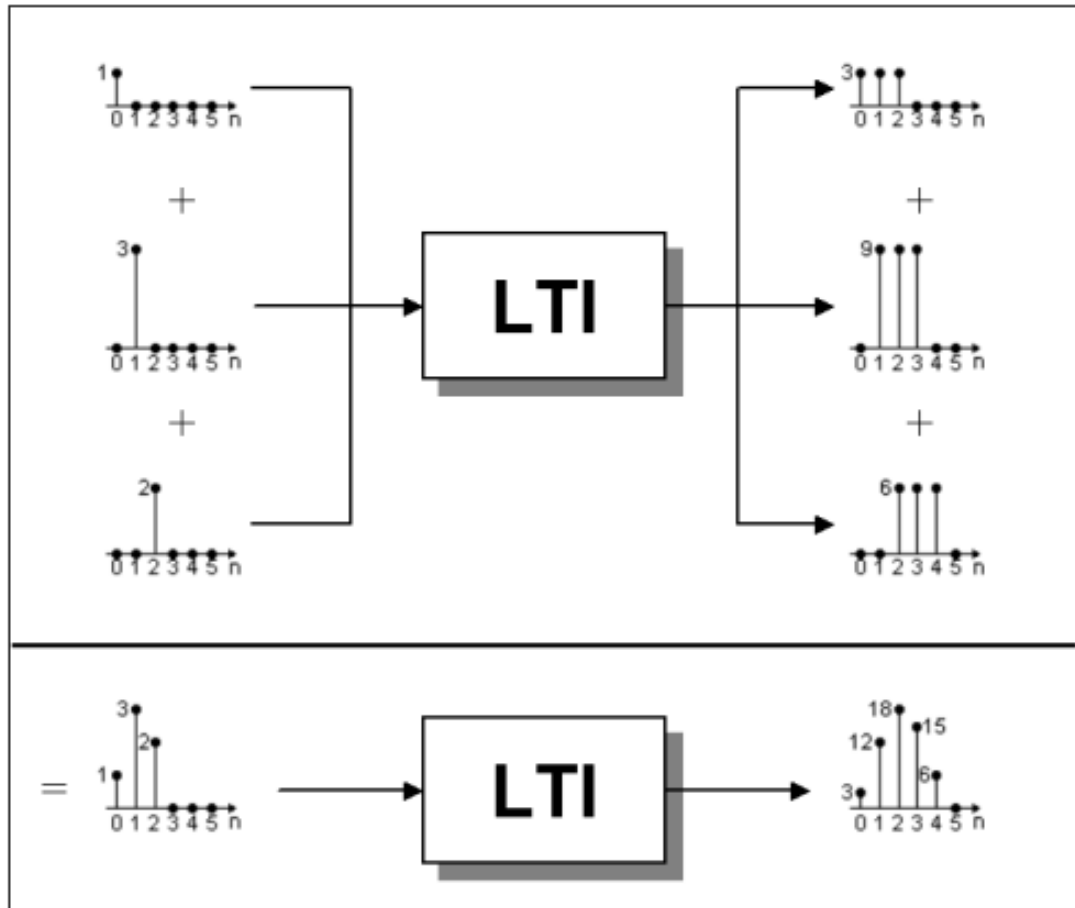


Figure 3.9: Convolution example [125].

$$\begin{aligned}
 x[n] &= \delta[n] \\
 y[n] &= x[n] * h[n] \\
 &= \delta[n] * h[n] \\
 &= h[n]
 \end{aligned}
 \tag{3.13}$$

We can model the room, loudspeaker, and microphone system as a Discrete-Time LTI system. Once we measure the impulse response of this system, we can compute the reverberation ($y[n]$) for any transmitted sound (e.g., music, speech) by convolving the transmitted signal ($x[n]$) with the Room Impulse Response (RIR) signal ($h[n]$). This implies that if we can accurately measure the Room Impulse Response, we can simulate how the sound would be perceived at that specific room, loudspeaker, and microphone location, as if we were physically present there [128].

Reciprocally, it has been found that when the Room Impulse Response (RIR) is measured at the exact same loudspeaker and microphone positions, inverting (deconvolving) the reverberation is feasible. This enables both enhancement and reduction of the reverberant energy present in recorded signals (e.g., speech) under similar conditions. However, if the Room Impulse Response recorded at a different position is used for deconvolution, it does not effectively remove the reverberation effect [129].

Being a Discrete-Time LTI system, the Room Impulse Response (RIR) $h[n]$ can be measured by transmitting a unit impulse $\delta[n]$ from the loudspeaker into the room as the input to the LTI system and recording the response at the microphone as the system's output, yielding $y[n] = \delta[n] * h[n] = h[n]$. Historically, unit impulses were physically generated using methods such as popping a balloon, firing a starter pistol, or hand clapping. However, these sound sources lacked repeatability. To improve consistency, researchers began using loudspeakers and considered the use of impulsive signals [130, 131], which directly estimate the impulse response of a linear system. Unfortunately, an impulse is an impractical excitation signal, as it theoretically has infinite amplitude and an infinitely short duration, posing a risk of damaging the loudspeaker or triggering the amplifier's protection circuit [132].

Due to these limitations, periodic signals (sequences) are used instead of a single impulse when measuring RIR. Impulses perform poorly in the presence of noise because they typically have very low energy. In contrast, white noise or periodic signals generally contain significantly more overall energy under the same amplitude constraints, leading to a much better signal-to-noise ratio (SNR) in measurements. While extracting the measurement signal can be more complex, particularly when phase information of the transfer function is required, it is still achievable. Common periodic signals used for RIR measurement include Maximum Length Sequence (MLS) [133], Time Delay Spectrometry (TDS) [134], and Exponential Sine Sweep (ESS) [135] [132]. In this thesis, we used the ESS signal to construct the GTU-RIR dataset.

A discrete-time signal $x[n]$ is periodic with period N if Equation (3.14) holds, where N is a positive integer. Sound waves, as waveforms, are inherently periodic signals. The fundamental period is the smallest positive integer N , and the fundamental frequency is given by $\omega_0 = 2\pi/N$ [125].

$$x[n] = x[n + N] \quad (3.14)$$

A complex exponential $e^{j(2\pi/N)n} = \cos((2\pi/N)n) + j\sin((2\pi/N)n)$ (Euler's formula) is periodic with period N . The set of all discrete-time complex exponential signals that are periodic with period N is given by Equation (3.15). All of these signals have fundamental frequencies that are multiples of $2\pi/N$ and thus are harmonically related, there are only N distinct signals in the set given in Equation (3.15). This is a consequence of the fact that discrete-time complex exponentials which differ in frequency by a multiple of 2π are identical [125].

$$\begin{aligned} \phi_k[n] &= e^{jk(2\pi/N)n} \\ &= e^{jk\omega_0 n}, k = 0, \pm 1, \pm 2, \pm 3, \dots \end{aligned}$$

$$\begin{aligned} \phi_0[n] &= \phi_N[n] \\ \phi_1[n] &= \phi_{N+1}[n] \\ &\vdots \\ \phi_k[n] &= \phi_{k+rN}[n] \end{aligned} \quad (3.15)$$

If we express the input signal as a linear combination of ϕ_k , we can represent the discrete-time signal $x[n]$ as shown in Equation (3.16). This representation is known as the discrete-time Fourier series, where the coefficients a_k given in Equation (3.19), are referred to as the Fourier series coefficients [125].

$$\begin{aligned} x[n] &= \sum_{k=\langle N \rangle} a_k \phi_k[n] \\ x[n] &= \sum_{k=\langle N \rangle} a_k e^{jk\omega_0 n} \\ x[n] &= \sum_{k=\langle N \rangle} a_k e^{jk(2\pi/N)n} \end{aligned} \quad (3.16)$$

$$X(e^{j\omega}) = \sum_{n=-\infty}^{+\infty} x[n] e^{-j\omega n} \quad (3.17)$$

$$\omega_0 = \frac{2\pi}{N} \quad (3.18)$$

$$\begin{aligned} a_k &= \frac{1}{N} \sum_{n=-\infty}^{+\infty} x[n] e^{-jk(2\pi/N)n} \\ &= \frac{1}{N} X(e^{jk\omega_0}) \end{aligned} \quad (3.19)$$

By substituting the term a_k into Equation (3.16), we obtain Equation (3.20). As $N \rightarrow \infty$ the fundamental frequency ω_0 , approaches zero, leading to the integral form represented in Equation (3.21) [125].

$$\begin{aligned}
x[n] &= \sum_{k=\langle N \rangle} \frac{1}{N} X(e^{jk\omega_0}) e^{jk\omega_0 n} \\
&= \sum_{k=\langle N \rangle} \frac{\omega_0}{2\pi} X(e^{jk\omega_0}) e^{jk\omega_0 n} \\
&= \frac{1}{2\pi} \sum_{k=\langle N \rangle} \omega_0 X(e^{jk\omega_0}) e^{jk\omega_0 n} \\
&= \frac{1}{2\pi} \sum_{k=\langle N \rangle} X(e^{jk\omega_0}) e^{jk\omega_0 n} \omega_0
\end{aligned} \tag{3.20}$$

$$x[n] = \frac{1}{2\pi} \int_{2\pi} X(e^{j\omega}) e^{j\omega n} d\omega \tag{3.21}$$

Finally, we obtain the equation pair 3.22, known as the discrete-time Fourier transform (DTFT) pair. The function $X(e^{j\omega})$ is referred to as the discrete-time Fourier transform and is also known as the spectrum of $x[n]$ since it provides information on how $x[n]$ is composed of complex exponentials at different frequencies [125]. The same transformation can be applied to $h[n]$ and $y[n]$ resulting in $H[k]$ and $Y[k]$. $H[k]$ is known as the transfer function of a linear time-invariant (LTI) system [125].

$$\begin{aligned}
x[n] &= \frac{1}{2\pi} \int_{2\pi} X(e^{j\omega}) e^{j\omega n} d\omega \\
X(e^{j\omega}) &= \sum_{n=-\infty}^{+\infty} x[n] e^{-j\omega n}
\end{aligned} \tag{3.22}$$

$$\begin{aligned}
X(e^{j\omega}) &= DTFT(x[n]) \\
x[n] &= DTFT(X(e^{j\omega})) \\
x[n] &\xleftrightarrow{DTFT} X(e^{j\omega})
\end{aligned} \tag{3.23}$$

The Discrete-Time Fourier Transform (DTFT) converts a time-domain signal $x[n]$ into its frequency-domain representation $X(e^{j\omega})$ or vice versa, as shown in Equation (3.23). The key properties of the DTFT can be summarized as follows [125] :

1. Precondition for all the following properties : $x[n] \xleftrightarrow{DTFT} X(e^{j\omega})$
2. Periodicity : $X(e^{j\omega}) = X(e^{j\omega+2\pi})$
3. Linessarity : if $x_1[n] \xleftrightarrow{DTFT} X_1(e^{j\omega})$ and $x_2[n] \xleftrightarrow{DTFT} X_2(e^{j\omega})$ then $ax_1[n] + bx_2[n] \xleftrightarrow{DTFT} aX_1(e^{j\omega}) + bX_2(e^{j\omega})$

4. Time Shifting and Frequency Shifting : $x[n - n_0] \xleftrightarrow{\mathcal{DTFT}} e^{-j\omega n_0} X(e^{j\omega})$ and $e^{j\omega_0 n} x[n] \xleftrightarrow{\mathcal{DTFT}} X(e^{j(\omega - \omega_0)})$
5. Conjugation and Conjugate Symmetry : $x^*[n] \xleftrightarrow{\mathcal{DTFT}} X^*(e^{-j\omega})$
6. Differencing and Accumulation : $x[n] - x[n - 1] \xleftrightarrow{\mathcal{DTFT}} (1 - e^{-j\omega}) X(e^{j\omega})$
7. Time Reversal : $x[-n] \xleftrightarrow{\mathcal{DTFT}} X(e^{-j\omega})$
8. Time Expansion : $if(n \% k = 0) x_{(k)}[n] = x[n/k] else x_{(k)}[n] = 0, x_{(k)}[n] \xleftrightarrow{\mathcal{DTFT}} X(e^{jk\omega})$
9. Differentiation in Frequency : $nx[n] \xleftrightarrow{\mathcal{DTFT}} j \frac{dX(e^{jk\omega})}{d\omega}$
10. Parseval's Relation : $\sum_{n=-\infty}^{+\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{2\pi} |X(e^{j\omega})|^2 d\omega$

Another crucial property of the Discrete-Time Fourier Transform (DTFT) relates to the convolution of signals. The output of a linear time-invariant (LTI) system can be expressed as a convolution, as shown in Equation (3.12), and its DTFT is given in Equation (3.24). This property is fundamental in extracting the room impulse response ($h[n]$) from the reverberated sound ($y[n]$).

$$Y(e^{j\omega}) = X(e^{j\omega})H(e^{j\omega}) \quad (3.24)$$

Room impulse response measurement was performed using quasi unit impulses such as popping a balloon, firing a starter pistol, or hand clapping. An impulse excitation is certainly simple: you get the directly the impulse response. The first reason to use an impulse is that the impulse response of an LTI system describes perfectly the system behaviour. The second reason is an impulse covers all frequencies when transformed into frequency domain as shown in Figure 3.10 which means that we are able to get the impulse response physically for all frequencies. When such a signal is passed through a system, the system shapes the impulse signal's spectrum by applying varying gains to each of the frequencies, thus acting as a filter. Mathematically this means that the system's impulse response in the frequency domain, is akin to that of a filter's frequency response. So naturally, to get the frequency response to an arbitrary input signal, we just need to multiply the input signal's frequency spectrum with the system's impulse response [125].

Room impulse response (RIR) measurement has traditionally been performed using quasi-unit impulses, such as popping a balloon, firing a starter pistol, or hand clapping. Using an impulse excitation is straightforward since it directly provides the impulse response.

There are two key reasons for using an impulse:

1. The impulse response of a linear time-invariant (LTI) system perfectly character-

izes the system's behavior.

2. An impulse contains all frequencies when transformed into the frequency domain, as shown in Figure 3.10, allowing us to capture the system's response across the entire frequency spectrum.

When such an impulse signal passes through a system, the system modifies its spectrum by applying different gains to each frequency component, effectively acting as a filter. Mathematically, this means that the system's impulse response in the frequency domain is analogous to a filter's frequency response. Consequently, to determine the frequency response of an arbitrary input signal, we simply multiply the input signal's frequency spectrum by the system's impulse response.

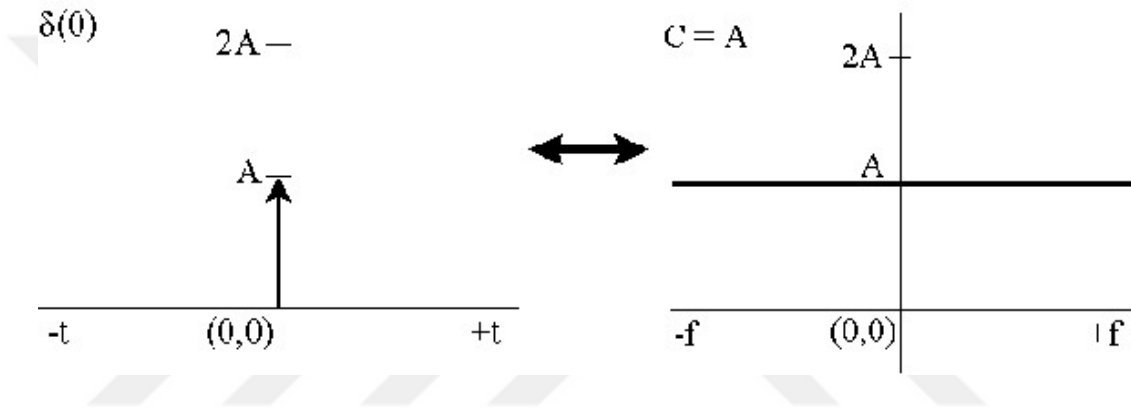


Figure 3.10: Single impulse contains all frequencies [125].

In practice, generating and replicating a perfect unit impulse signal is extremely challenging. Additionally, impulses perform poorly in noisy environments because they generally have very low energy, making them susceptible to noise interference. Compared to impulses, white noise—under the same amplitude constraints—has significantly higher overall energy, leading to a much better signal-to-noise ratio (SNR) in measurements. As a result, instead of using impulse excitation, we transmit Maximum Length Sequence (MLS) or Exponential Sine Sweep (ESS) signals and then deconvolve them with the reverberated signal to obtain the room impulse response (RIR) [136].

To obtain an impulse response of a system, it is more convenient to convert time domain signals to frequency domain signals using DTFT, make necessary calculations and convert back to time domain signals using inverse DTFT (IDTFT). This process is called deconvolution of $h[n]$ and $x[n]$, and summarized in Equation (3.25). Time and frequency domain conversions are described in Figure 3.11 and Equation (3.22) [137].

$$\text{impulse response} = h[n] = IDTFT\left(\frac{DTFT(y[n])}{DTFT(x[n])}\right) = IDTFT\left(\frac{Y(e^{j\omega})}{X(e^{j\omega})}\right) \quad (3.25)$$

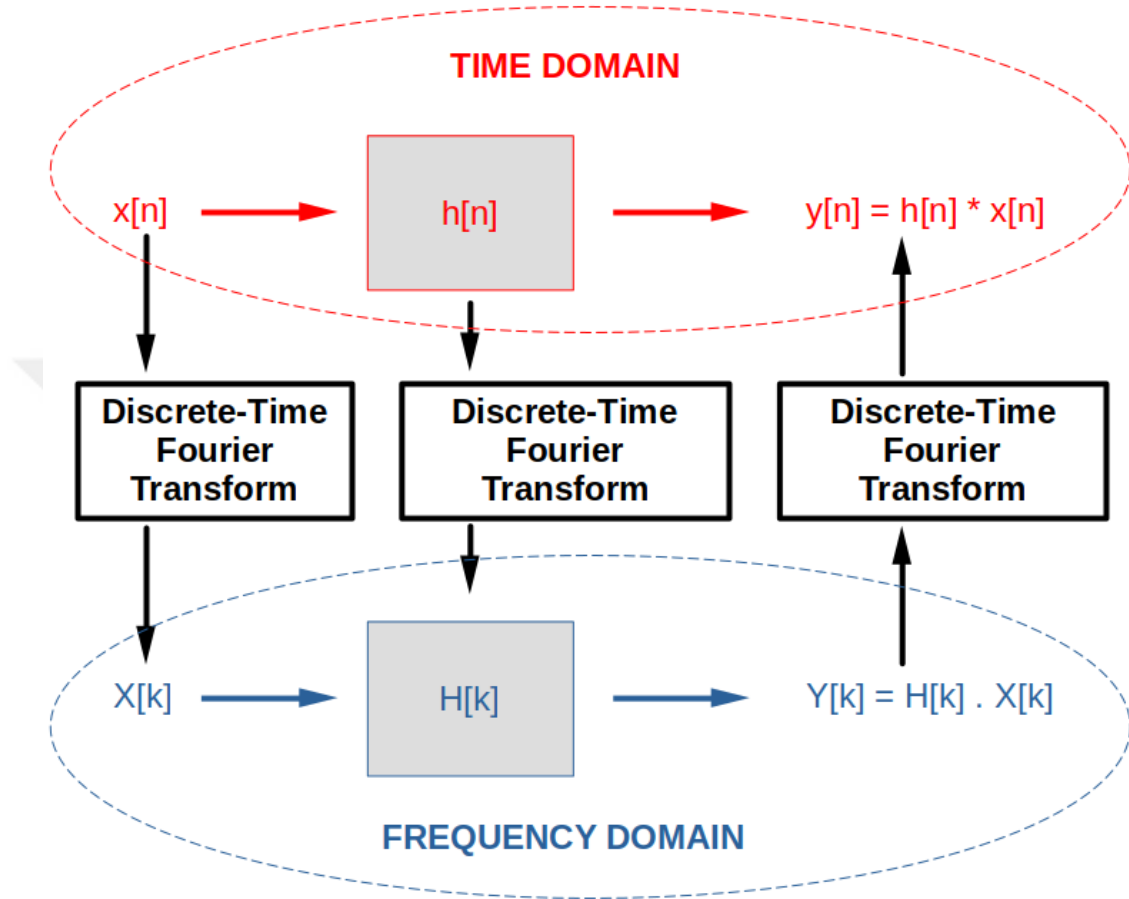


Figure 3.11: Discrete Time Fourier Transform. [138]

This approach, however, is sensitive to certain challenges [137]:

- Nulls in $X[k]$: A divide-by-zero error occurs if $X[k]$ has zeros at any frequency. Even small values of $X[k]$ near nulls can amplify noise from $X[k]$, resulting in poor accuracy.
- Signal Design: An input signal with an DTFT designed to avoid nulls or low energy spots across the frequency range minimizes these issues.

The impulse response deconvolution process can also be performed by linearly convolving the measured output with an analytical inverse filter derived from the excitation signal. Using linear convolution helps avoid time-aliasing issues. Rather than directly deconvolving $y[n]$ and $x[n]$, as shown in Equation (3.25), a more effective approach is

to construct the inverse filter using $x[n]$ and then convolve it with $y[n]$. Mathematically, this yields the same result as in Equation (3.25) [139].

Theoretically, the process can be described step by step as follows [136]:

1. Transmit an Exponential Sine Sweep (ESS) signal into the room using a loud-speaker. This transmitted signal is theoretically denoted as $x[n]$.
2. Receive the reverberated signal, which is theoretically denoted as $y[n]$.
3. The goal is to obtain $h[n]$, the room impulse response, from the equation $y[n] = x[n] * h[n]$
4. To achieve this, we need to deconvolve $y[n]$ and $x[n]$ to extract $h[n]$
5. Instead of directly performing deconvolution, we construct the inverse filter of $x[n]$, denoted as $x'[n]$, and convolve it with $y[n]$,
6. This operation results in $h[n]$, expressed as $h[n] = y[n] * x'[n]$

The inverse filter for the Exponential Sine Sweep (ESS) signal is constructed using Equation (3.26), where T represents the sweep duration, f_1 is the starting frequency, and f_2 the ending frequency. The transmitted ESS signal $x[n]$, spans T seconds.

For our implementation, we selected $f_1 = 15$ Hz and $f_2 = 22000$ Hz to ensure that the inverse filter fully covers the ESS signal's frequency range, which extends from 20 Hz to 20 kHz. This range is chosen to align with the human hearing range (20 Hz – 20 kHz), though most people typically perceive frequencies between 200 Hz and 12 kHz [136].

An example of an ESS signal and its corresponding inverse filter is shown in Figure 3.12. To efficiently implement the Discrete-Time Fourier Transform (DTFT), we utilize the Fast Fourier Transform (FFT) algorithm [140].

$$x'[n] = \frac{x[n]}{e^{\frac{n \ln(\frac{f_1}{f_2})}{T}}} \quad (3.26)$$

3.1.2. Practical Explanation of RIR Measurement

A unit impulse can be used to measure the room impulse response (RIR), but its total energy is very limited, making it sensitive to noise and difficult to reproduce accurately. A more effective approach is to use white noise, a random signal with equal intensity across all frequencies, resulting in a constant power spectral density. A unit impulse covers all frequencies simultaneously, whereas white noise achieves this by containing a large number of equally distributed frequencies. White noise covers the human hearing range (20 Hz to 20 kHz) and provides a uniform energy distribution, ensuring all

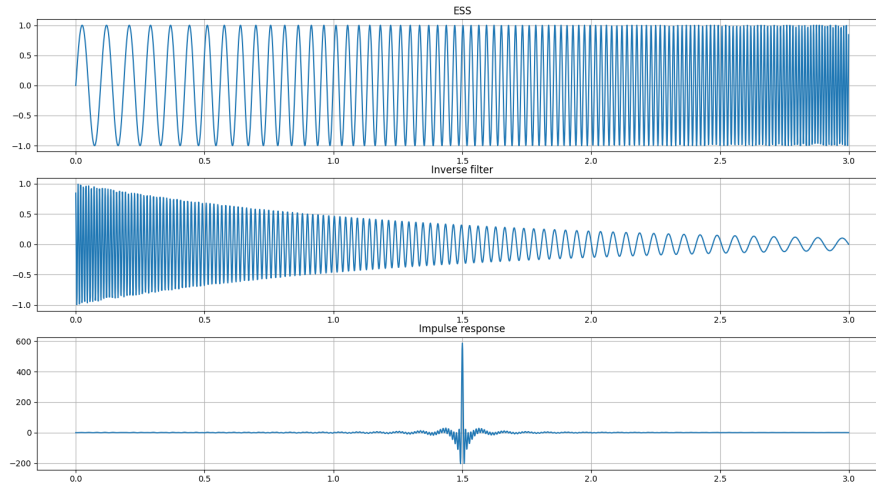


Figure 3.12: Example Inverse Filter [141].

frequencies contribute equally to the measurement. Its random nature helps minimize distortions and improve the signal-to-noise ratio (SNR). In practice, a finite-duration white noise signal is used, meaning frequencies within the 20 Hz–20 kHz range are randomly distributed over a limited time interval. This allows for accurate measurement of how sound propagates in a room [142].

The reason that we don't use single frequency sound signal (e.g. only 440Hz sound signal, which is note A) is that each material in a room absorbs different frequency range of sounds [143, 144] as shown in Table 3.2. It can be seen in Figure 3.13 that reverberation time for 60DB (RT60) values for each frequency band is different. [145]. To cover impulse responses for all frequencies, we have to transmit a white noise, a pink noise or a sine sweep signal.

Table 3.2: Sound absorption coefficients for common materials at 500 Hz, 1000 Hz, and 2000 Hz [146].

Material Type	Material	Coefficient 500Mhz	Coefficient 1000Mhz	Coefficient 2000Mhz
Wall	Brick, unglazed	0.03	0.04	0.05
Wall	Brick, unglazed,painted	0.02	0.02	0.02

Table 3.2: (continued) Sound absorption coefficients for common materials at 500 Hz, 1000 Hz, and 2000 Hz [146].

Material Type	Material	Coefficient 500Mhz	Coefficient 1000Mhz	Coefficient 2000Mhz
Wall	Plaster, gypsum, or lime, smooth finish on tile or brick	0.02	0.03	0.04
Wall	Plaster, gypsum, or lime, rough or smooth finish on lath	0.06	0.05	0.04
Wall	Concrete block, light, porous	0.31	0.29	0.39
Wall	Concrete block, dense, painted	0.06	0.07	0.09
Wall	Gypsum boards, 1/2-inch nailed to 2x4s, 16 inches o.c.	0.05	0.04	0.07
Wall	Plywood paneling, 3/8-inch thick	0.17	0.09	0.10
Wall	Large panes of heavy plate glass	0.04	0.03	0.02
Wall	Ordinary window glass	0.18	0.12	0.07
Floor	Concrete or Terazzo	0.015	0.02	0.02
Floor	Linoleum, asphalt, rubber, or cork tile on concrete	0.03	0.03	0.03
Floor	Wood	0.10	0.07	0.06
Floor	Wood parquet in asphalt on concrete	0.07	0.06	0.06
Floor	Carpet	0.14	0.37	0.60
Floor	Marble or glaze tile	0.01	0.01	0.02
Fabrics	Light velour, 10 oz per sq yd, hung straight, in contact with wall	0.11	0.17	0.24

Table 3.2: (continued) Sound absorption coefficients for common materials at 500 Hz, 1000 Hz, and 2000 Hz [146].

Material Type	Material	Coefficient 500Mhz	Coefficient 1000Mhz	Coefficient 2000Mhz
Fabrics	Medium velour, 14 oz per sq yd, drapped to half area	0.49	0.75	0.70
Fabrics	Heavy velour, 18 oz per sq yd, drapped to half area	0.55	0.72	0.70
Misc	Chairs, metal or wood seats,each,unoccupied	0.22	0.39	0.38

Other than white noise, different types of test impulse signals have been developed to optimize impulse response measurements by ensuring good immunity to background noise and enabling straightforward deconvolution. These include:

- Maximum Lenght Sequence (MLS), also known as "pseudo-random white noise"
- Time Delay Spectrometry (TDS), also known as "linear sine sweep", "stretched pulse", "chirp"
- Exponential Sine Sweep (ESS)

Each of these test signals can be combined with various deconvolution techniques, resulting in diverse measurement methods. Among these, the Exponential Sine Sweep (ESS) with non-circular deconvolution [147] is the preferred approach due to its theoretical robustness and practical effectiveness in handling noise and nonlinearity. ESS signal properties are described in Chapter 2.

Despite the effectiveness of these techniques, measuring room impulse responses (RIRs) for complex environments remains challenging. For any given room geometry, surface reflection coefficients, speaker locations, and microphone positions, impulse response measurements are specific to each unique combination. Consequently, sampling a volume of interest requires a large number of spatial sampling points, which introduces additional complexities, such as ensuring precise microphone positioning to capture accurate RIRs and addressing calibration challenges to maintain consistency across multiple microphones.

In this study, because of these challenges we conducted limited number of RIR measurements for 11 distinct rooms within the Computer Engineering Department at Gebze

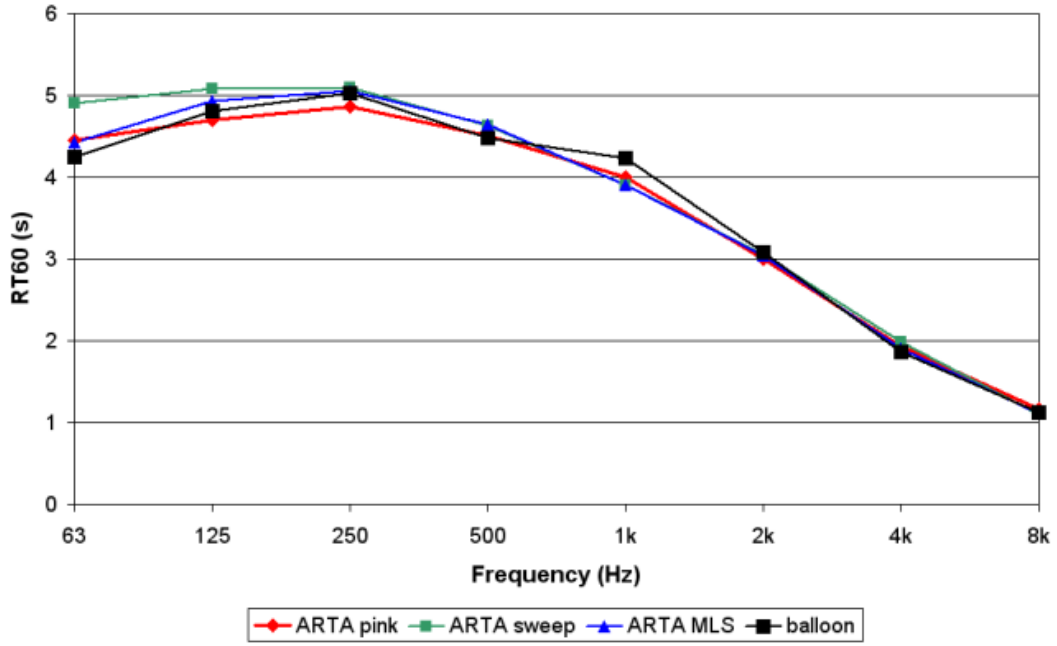


Figure 3.13: Example RT60 values for various frequency bands measured using Pink Noise, ESS, MLS, and a balloon popping[145].

Technical University. This dataset of real RIRs will serve as a benchmark to evaluate the performance and accuracy of our RIR generation algorithms.

3.2. GTU-RIR Data Collection System

The system used for data collection consists of a microphone stand, a loudspeaker stand, and a computer, as illustrated in Figure 3.14. Both the microphone and loudspeaker stands are designed to rotate around their respective Z-axes, allowing precise positioning for capturing reverberated sound data. A photograph of the setup during data recording is shown in Figure 3.15.

The collected dataset, along with a detailed explanation of the system’s design and operation, as well as all the scripts used during the data collection process, are publicly available in the ”gtu-rir” GitHub repository¹. This ensures transparency and facilitates reproducibility for researchers aiming to utilize or build upon this work.

A single computer is connected to all the devices in the system, including loudspeakers, microphones, stepper motor controllers, and a webcam, to facilitate the Room Impulse Response (RIR) data collection process. The reverberated sound data is captured

¹<https://github.com/mehmetpekmezci/gtu-rir>

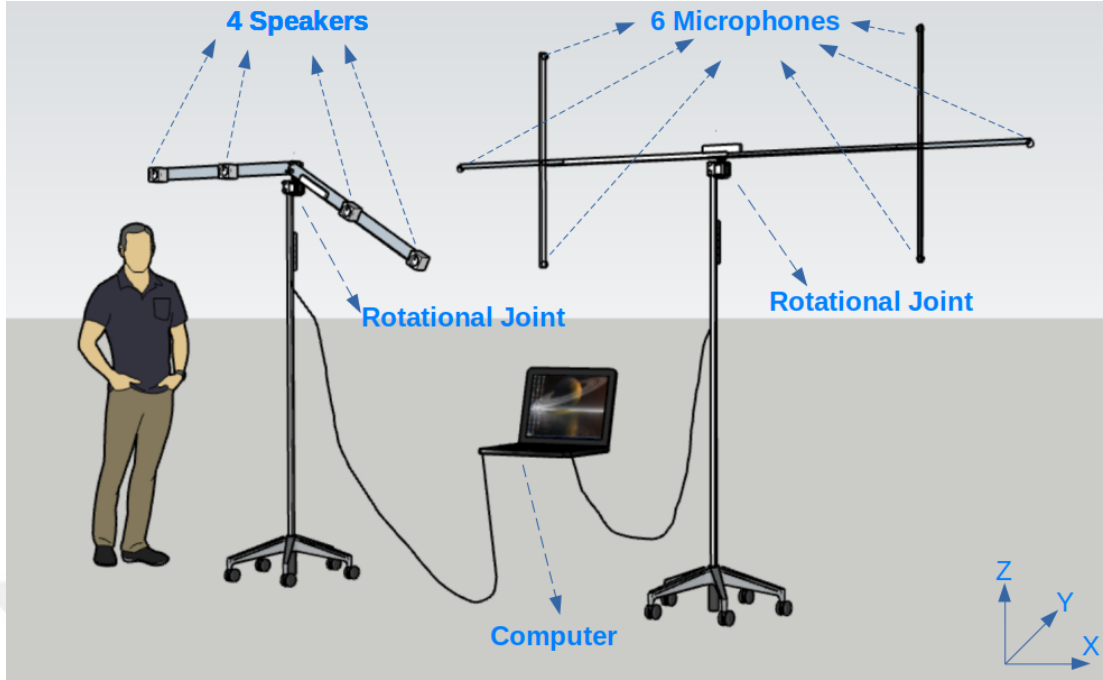


Figure 3.14: Illustration of RIR data collection mechanism used to collect GTU-RIR dataset.

using Algorithm 1, which automates the recording process. To ensure minimal noise interference, data sampling is scheduled between 21:00 and 08:00 through automation routines.

During the recording process, stepper motors control the rotation of the loudspeaker and microphone stands, ensuring precise positioning at each phase. A webcam is used to capture images of the system at different stages, enabling the detection of potential mechanical issues.

Before the start of each recording session, automation routines conduct comprehensive checks on all connected devices to confirm their operational status. These routines also prompt for manual inspections, including verifying the positioning of the webcam and the input of coordinates for the microphone and loudspeaker stands as well as the dimensions of the room. This structured workflow ensures consistent and reliable RIR data collection.

The recordings use an Exponential Sine Sweep (ESS) signal with 2 seconds of pre- and post-silence. The ESS signal is generated following Farina's algorithm [135] and comprises waves ranging from 60 Hz to 15 KHz. The total duration of the ESS signal is 10 seconds, with a sampling rate of 44,100 Hz, ensuring high-quality and precise data capture.



Figure 3.15: The actual RIR data collection system used to collect GTU-RIR dataset.

After recording the data, we manually inspect the webcam images to identify and delete any records affected by mechanical failures. Additionally, records with a negative signal-to-noise ratio (SNR) are discarded. Noise is removed by analyzing and sampling the frequencies during the 4-second silence period embedded in the transmitted ESS signal.

Once the data is cleaned, the Room Impulse Response (RIR) is extracted from the transmitted and received signals for each valid record, ensuring that only high-quality and accurate data is retained for further analysis.

3.2.1. Mechanical Construction

The GTU-RIR real RIR data collection system comprises a loudspeaker stand with four speakers, a microphone stand with six microphones, and a computer for control and data processing. While not all construction details are covered here, we highlight key aspects that enhance the system's robustness and ease of use. Before initiating the construction process, it is essential to gather the required materials, as outlined in Table 3.3.

Algorithm 1 RIR_DATA_COLLECTION

```
1: function RIR_Data_Collection
2:   for each microphone rotational position do
3:     for each speaker rotational position do
4:       for each of 4 speakers do
5:         Start_Listening_From_6_Microphones()
6:         Transmit_ESS_Signal_From_Selected_Speaker()
7:         Stop_Listening_From_6_Microphones()
8:         Take_a_Picture_of_Scene_with_Webcam()
9:         Save_All_Information()      ▷ recorded wav files, coordinates,
scene picture
10:      end for
11:      Rotate_Speaker_Stand_Using_Step_Motor()
12:    end for
13:    Rotate_Microphone_Stand_Using_Step_Motor()
14:  end for
15: end function
```

Table 3.3: Bill of materials.

Part Number	Description	Minimum Quantity
1	NEMA-23 Step Motor	2
2	TIC T500 Step Motor Driver Card (some pins should be soldered)	1
3	TIC T825 Step Motor Driver Card (some pins should be soldered)	1
4	Step Motor Power Adapter - 12V - 2A	2
5	USB - MicroUSB Cable 5-pin-male-data-charge-cable-cord-30cm	2
6	Metal Stand Pole 20mm x 20mm Square Stainless Steel or Aluminium Profile, 2 meters long	2
7	Desk chair base with casters	2
8	USB Hub 10 port	2

Table 3.3: (continued) Bill of materials.

Part Number	Description	Minimum Quantity
9	USB Hub Power Adapter 5V 1A	2
10	USB Lavalier Mini Microphone	10
11	5 way plug sockets with 10 meters extension cable	3
12	USB 5 meters extension cable	6
13	USB 1 meter extension cable	3
14	USB Webcam	1
15	Mini USB Speaker , 100Hz - 20KHz	2
16	Laptop Computer with 3 USB input, i3 cpu, 8GB Ram	1
17	Aluminium Angle L Profile 20mm x 60mm , 1 meter	10
18	M4 Steel Bolts and Nuts	20
19	M6 Steel Bolts and Nuts	20
20	M8 Steel Bolts and Nuts	20
21	1 Meter 8mm Lead Screw	2
22	Sellotape 24mm x 50m	4

The system boasts the following key properties to ensure robust functionality and efficiency during the data collection process:

- **Automated Testing:** The system automatically tests all microphones and speakers before starting a session to minimize faulty recordings.
- **Noise-Free Recording:** Sound is not recorded while the stands rotate to a new position, and only one speaker is active at a time during recording.
- **Mobility:** The stands are mounted on desk chair bases with casters, allowing easy movement across the room, as seen in Figures 3.18 and 3.23.
- **High Throughput:** The system can record up to 2400 data points per room automatically by combining 6 microphones and 4 speakers with 10 rotations of the loudspeaker stand per 10 microphone stand positions. This setup allows for unattended data collection sessions.
- **Step Motor Placement:** The step motor is mounted on the top of the stand poles, ensuring that only the aluminum profiles carrying the microphones and speakers

rotate. This design prevents interference from furniture shorter than 1.5 meters in height, as seen in Figure 3.17.

- **Lightweight Components:** The speakers, microphones, and aluminum profiles are lightweight, enabling easy control by the step motor. Assembly details can be seen in Figures 3.19, 3.20, 3.22, and 3.23.
- **Directional Guidance:** A green sticker on one side of the aluminum profiles helps determine the starting direction, which is always oriented toward the room's door. Additionally, a user guide script assists during the setup, as shown in Figure 3.16.
- **Error Handling:** Faulty records caused by USB cable errors or step motor movement issues are detected and discarded during data cleaning. Scripts monitor and resolve step motor angular errors.
- **USB Port Management:** Stickers with unique numbers are placed on USB hubs, microphone connectors, speaker connectors, and their respective mounting locations to maintain proper USB port assignments, as shown in Figure 3.24.
- **Photo Monitoring:** A connected webcam captures photos at each recording step, which are saved to the session directory. These photos help identify and discard records affected by mechanical problems.
- **Step Motor Calibration:** In cases where the step motor does not rotate to the correct angle (36 degrees per step), the issue is addressed using a script to monitor and adjust its position.
- **Coordinate Validation:** Speaker and microphone positions are converted to Cartesian coordinates and geometrically validated against previous values and probable region calculations to ensure accuracy.
- **Data Format:** All data is stored as lists of lists in the Python-compatible pickle file format, ensuring consistency and easy access.
- **Noise Sampling:** Two seconds of silence are added before and after the transmitted ESS signal, allowing noise frequencies to be sampled and subsequently removed from the recorded data.

These features collectively make the system highly reliable, user-friendly, and capable of efficiently capturing high-quality RIR data.



Figure 3.16: Microphone and loud speaker stand direction guide within the script.

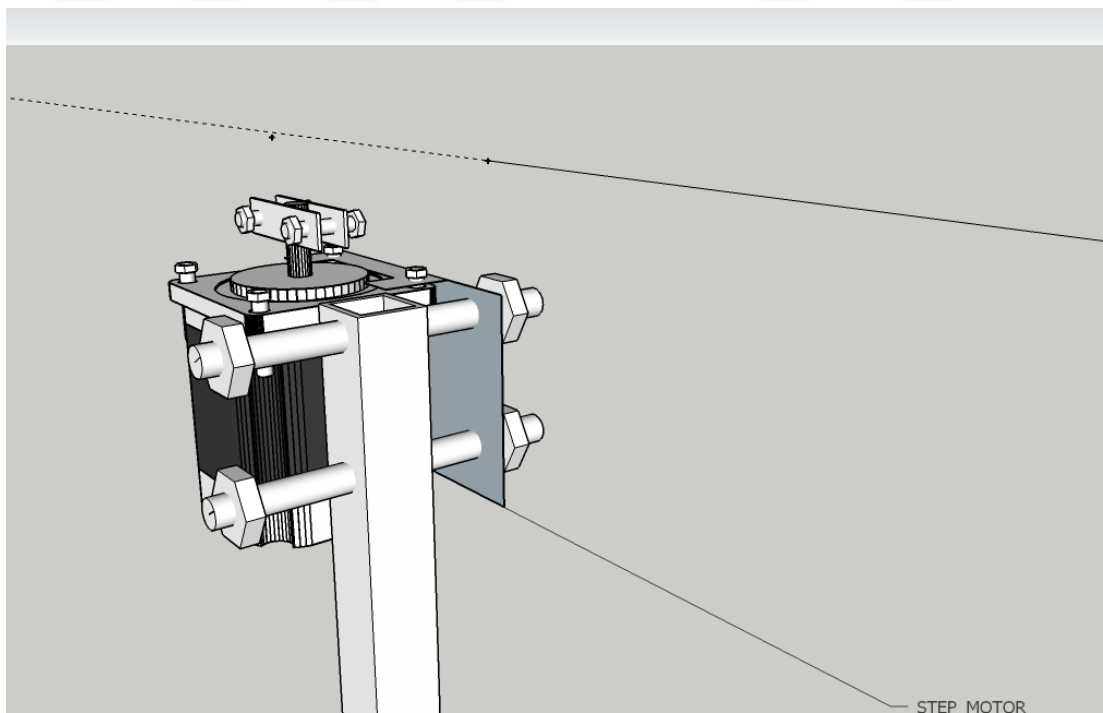


Figure 3.17: Step motor mounting on the stand pole.



Figure 3.18: Loud speaker stand.

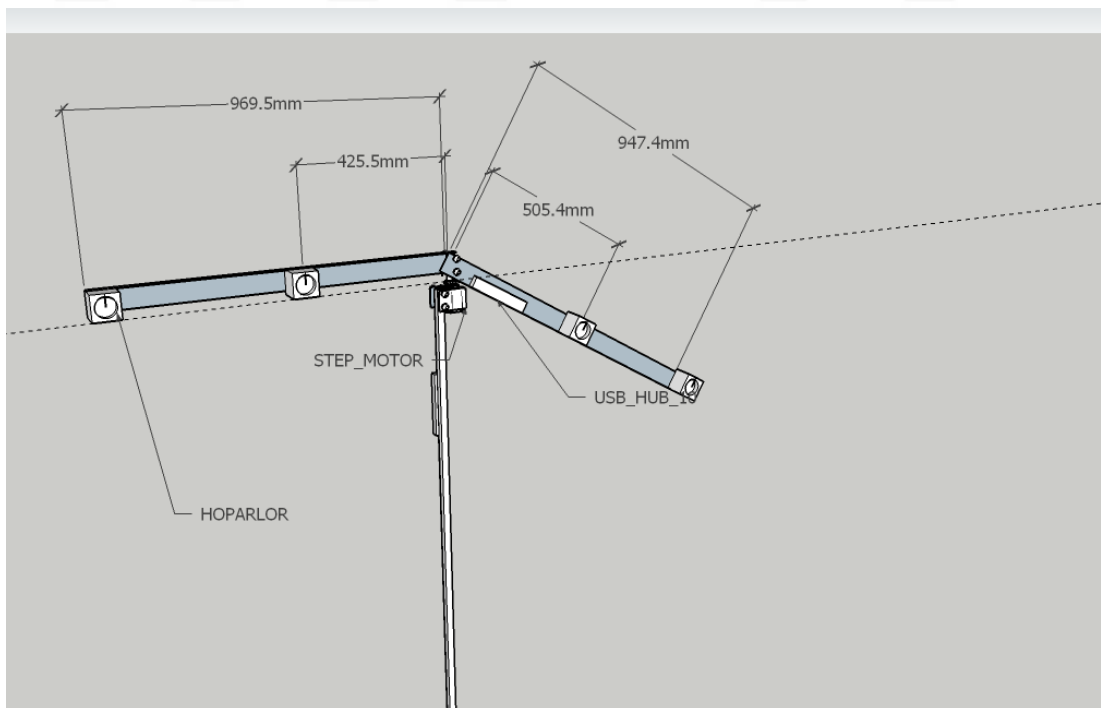


Figure 3.19: Loud speaker stand front.

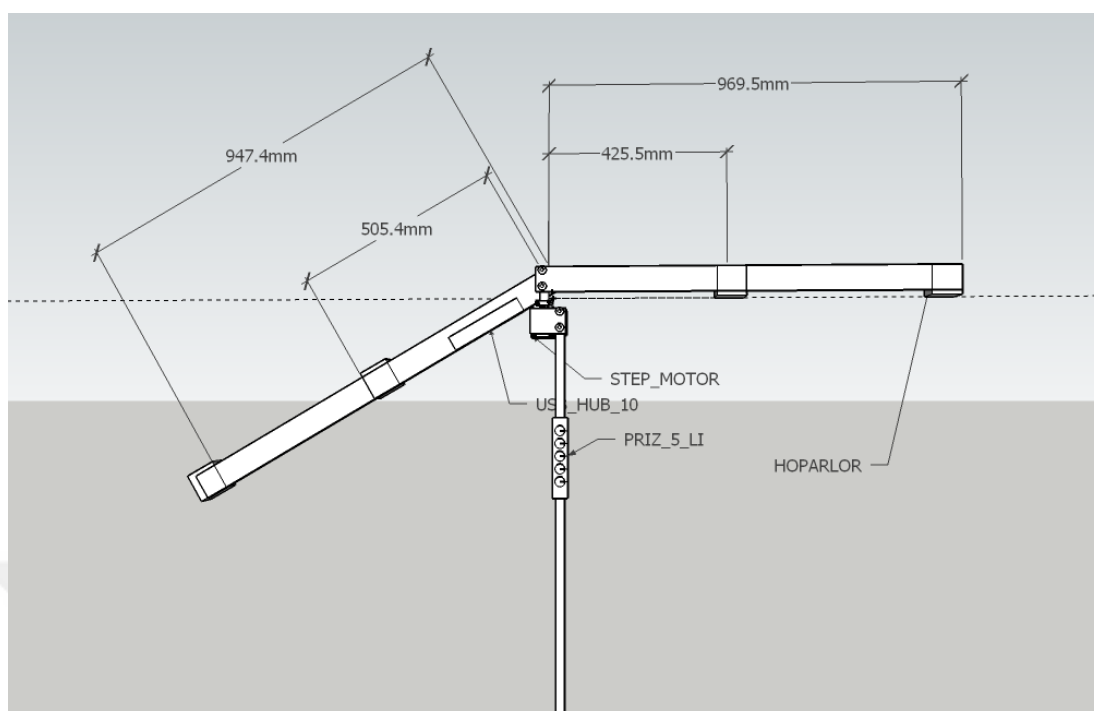


Figure 3.20: Loud speaker stand back.

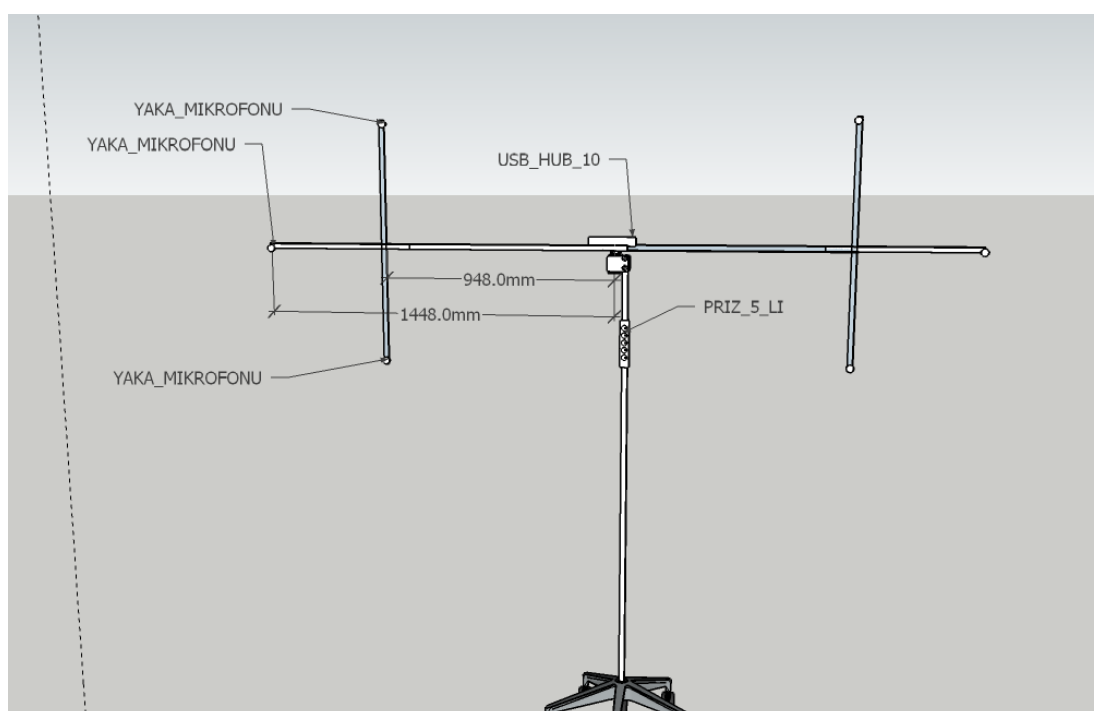


Figure 3.21: Microphone stand.

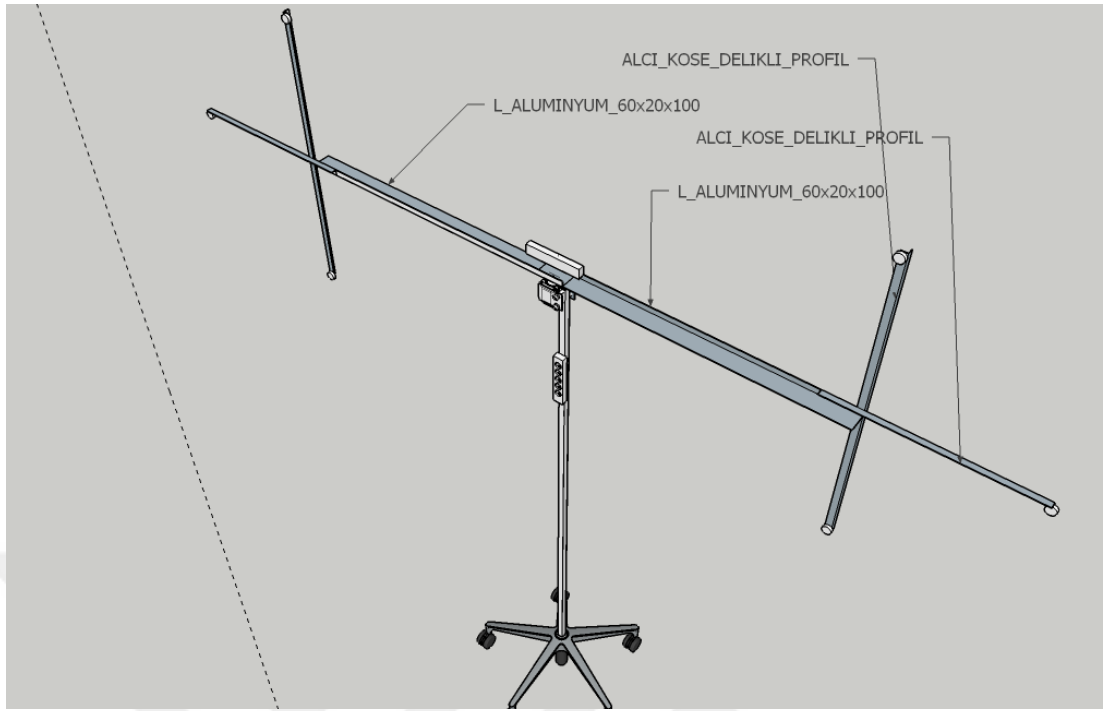


Figure 3.22: Microphone stand front.

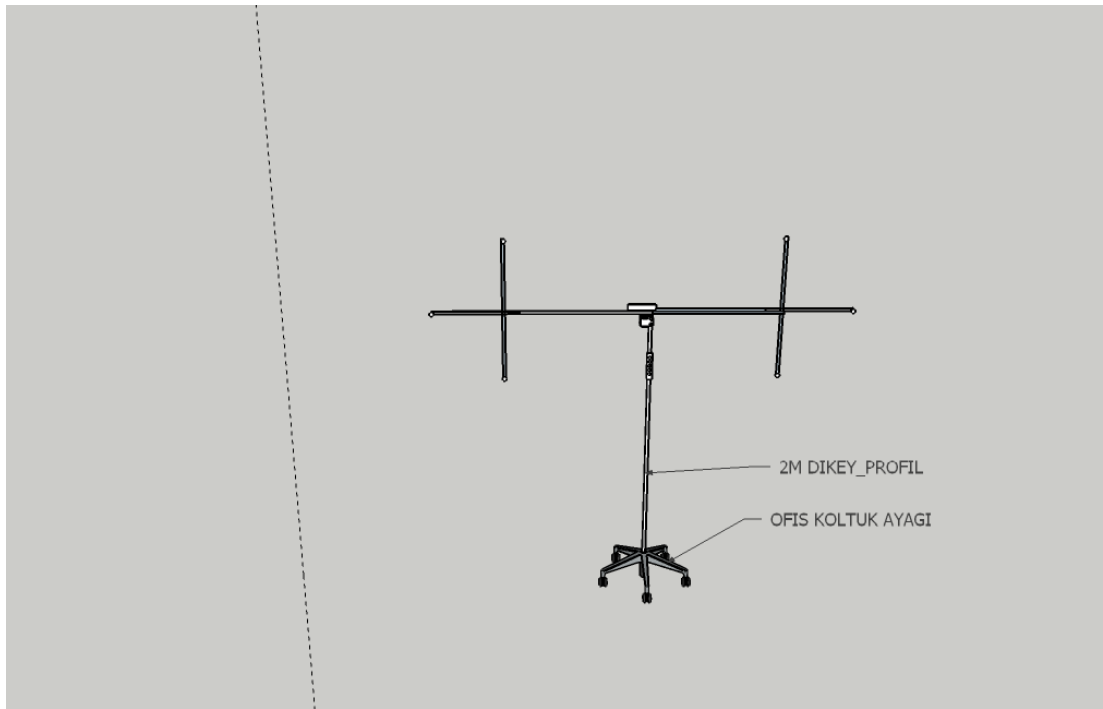


Figure 3.23: Microphone stand back.



Figure 3.24: Microphone stand number stickers on USB hub and USB devices.

3.2.2. Electrical Construction

All the loudspeakers, microphones, and step motors are connected to USB hubs located on their respective stands, as depicted in Figures 3.25, 3.26, 3.27, 3.28, and 3.29. These USB hubs are linked to the computer via five-meter USB cables. It is important to note that USB cables longer than five meters cause instability during sound recording. Therefore, the maximum permissible distance between the microphone and speaker stands is 10 meters, as both stands must remain connected to the same computer.

To ensure stability and consistent hardware recognition, the USB hubs are designed to retain the same hardware address, verified using the `lsusb` command. Additionally, the USB cables from the microphone and speaker stands must always be connected to the same designated ports on the computer. To maintain this consistency, stickers are placed on the computer's USB ports to indicate the specific connections for the microphone and speaker stand USB hubs. This systematic approach minimizes setup errors and ensures reliable operation during data collection.

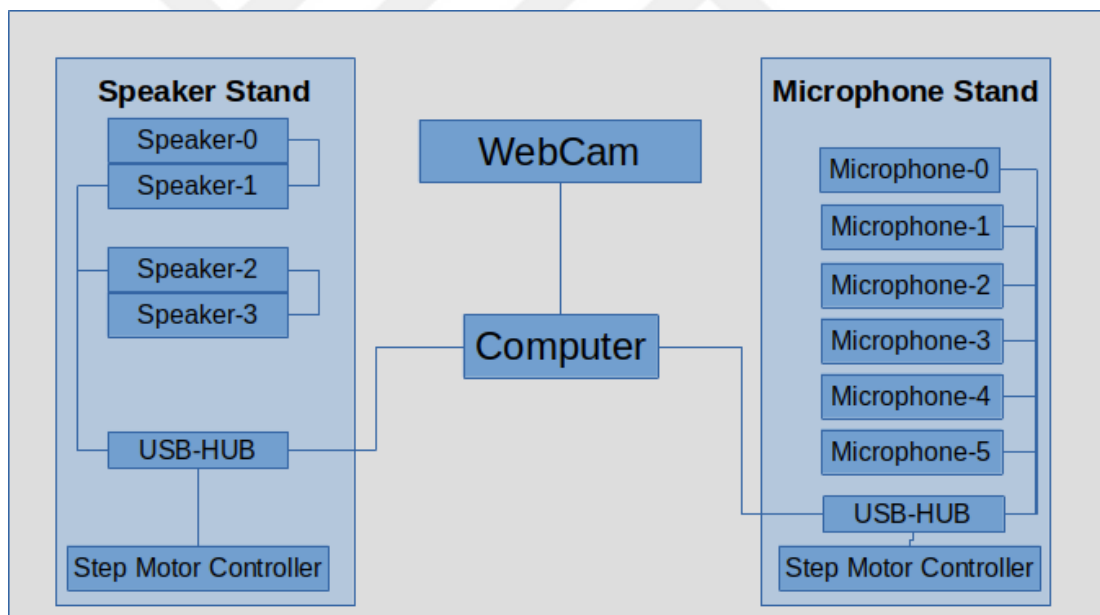


Figure 3.25: USB connections.

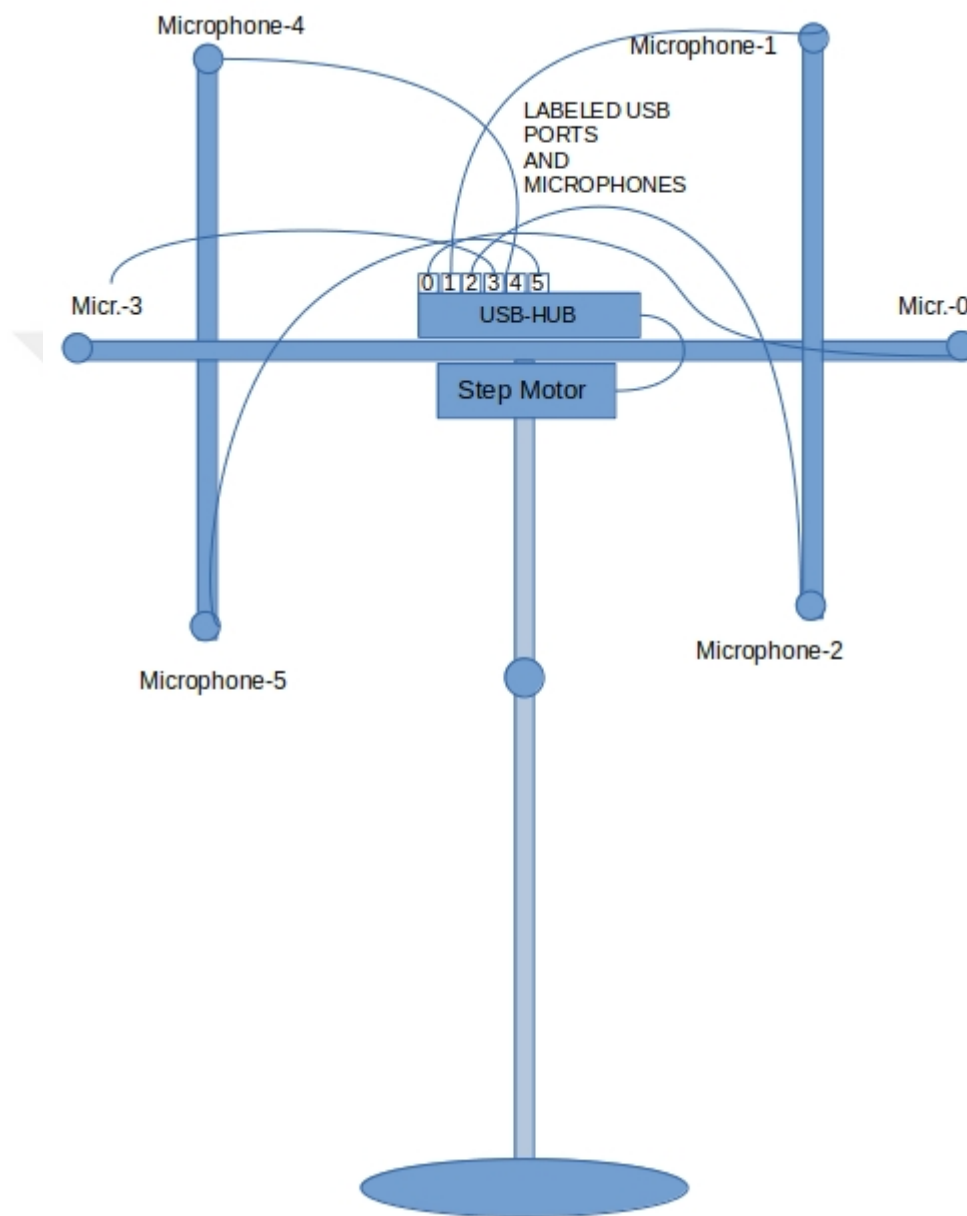


Figure 3.26: Microphone stand.



Figure 3.27: Microphone stand cables.

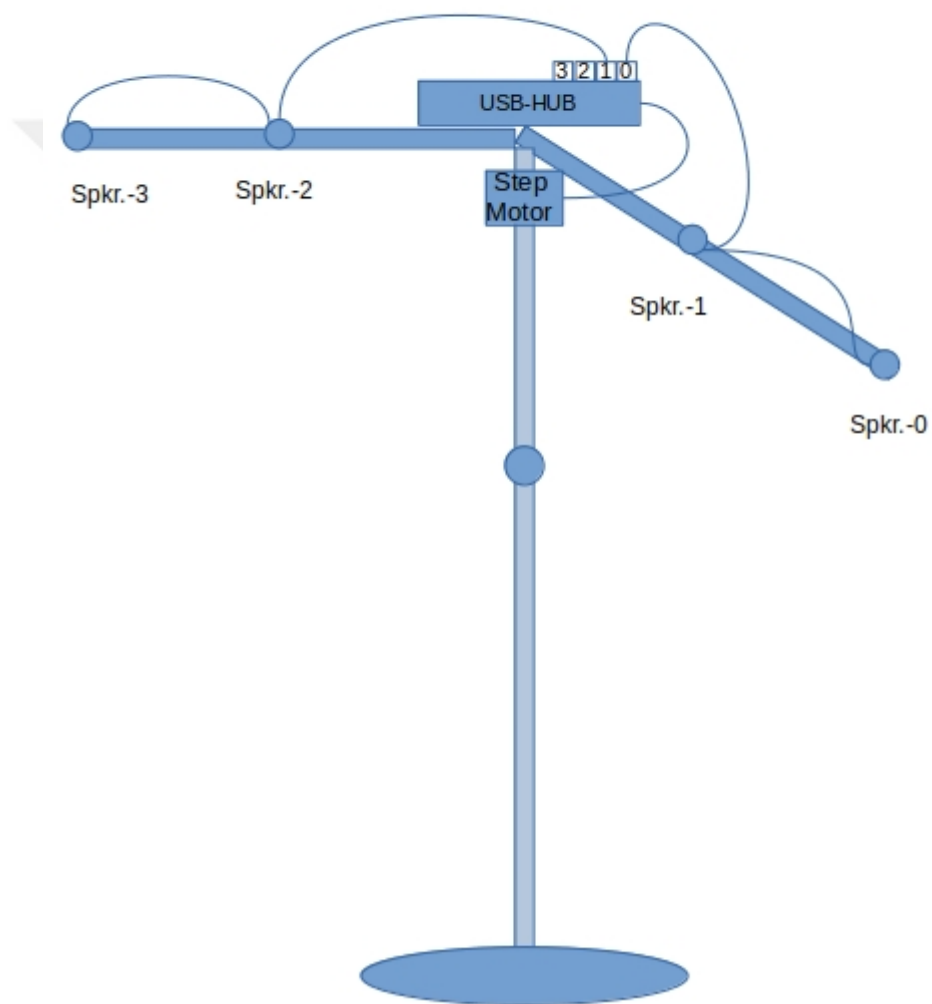


Figure 3.28: Loud seaker stand.

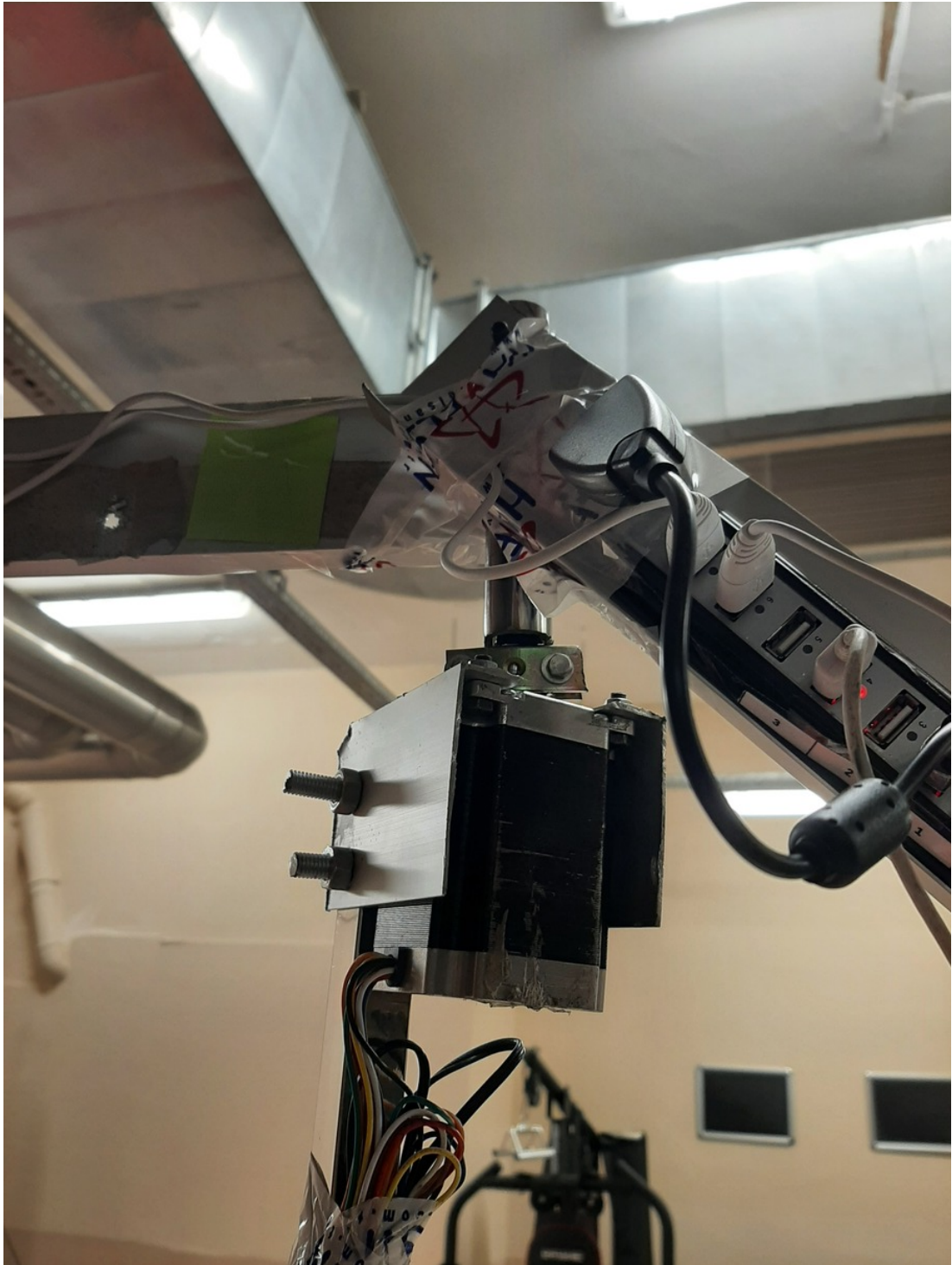


Figure 3.29: Loud speaker stand.

3.2.3. GTU-RIR Data Presentation

DATASET	Room	Vol. (m^3)	Surf. (m^2)	Smp.	RT60	Spk- Mic (m)	Type
GTU-RIR	207	172.8	216.6	1441	0.6	5.9	Class
GTU-RIR	208	146.1	186.5	1407	0.6	3.9	Class
GTU-RIR	Confer.	209.7	247.5	1471	0.6	4.3	Confr
GTU-RIR	Sport01	363.4	339.0	1576	1.1	3.0	Sport
GTU-RIR	Sport02	160.1	195.6	1371	1.0	3.9	Sport
GTU-RIR	Z02	139.7	173.1	1468	0.9	3.8	Class
GTU-RIR	Z04	139.9	173.4	1474	0.8	4.0	Class
GTU-RIR	Z06	326.5	330.7	1503	0.8	4.4	Class
GTU-RIR	Z10	328.6	334.6	1422	0.8	5.7	Class
GTU-RIR	Z11	204.6	234.0	959	0.8	4.6	Class
GTU-RIR	Z23	410.2	420.9	1110	0.5	7.6	Confr
	MEAN	236.5	259.26	1382	0.7	4.6	
BUT Re- verbDB	Confer.	1038.	884.7	123	1.2	14.4	Confr
BUT Re- verbDB	R.212	33.17	63.50	155	1.1	2.4	Hotel
BUT Re- verbDB	C236	102.1	136.2	310	1.1	3.0	Class
BUT Re- verbDB	D105	2704.	1335.	186	1.1	7.9	Confr
BUT Re- verbDB	E112	1104.	763.1	62	1.3	5.8	Confr
BUT Re- verbDB	L207	99.50	135.5	279	0.7	1.7	Office
BUT Re- verbDB	L212	107.6	144.6	155	0.8	2.1	Office
BUT Re- verbDB	L227	229.0	282.3	127	1.9	1.8	Stairs
BUT Re- verbDB	Q301	194.2	241.6	92	1.0	0.7	Class
	MEAN	623.5	343.1	165	1.13	4.4	

Table 3.4: (continued)

DATASET	Room	Vol. (m^3)	Surf. (m^2)	Smp.	RT60	Spk- Mic (m)	Type
---------	------	-------------------	--------------------	------	------	------------------------	------

Table 3.4: Statistical tables comparing the GTU-RIR and BUT ReverbDB data sets.

The volume of the room is given in the column Vol. The surface area of the entire room is given in the Surf. The Smp. column indicates the number of RIR samples collected in the corresponding room. The mode of the RT60 values in a room is given in the RT60 column. The mode of the loudspeaker-microphone distance values is given in the Spk-Mic column. The type of room is given in the Type column, i.e. a classroom (Class), a conference hall (Confr), a hotel room (Hotel), an office (Office), a staircase (Stairs) or a sports hall (Sport).

The RIRs in this dataset have a sampling rate of 44100 Hz and a duration of 2 seconds. All data are recorded in 11 different rooms of Gebze Technical University (GTU) as shown in the Figure 3.30 with volumes ranging from 140 m^3 to 410 m^3 .

The Room Impulse Response (RIR) data file contains a comprehensive dataset with over 15,000 data points. Each data point is represented as a list containing 43 elements. The data file can be accessed via the download link provided in the GTU-RIR GitHub repository ¹.

The following features are extracted from the dataset for analysis purposes:

1. Loudspeaker x, y, z positions: Specifies the 3D coordinates of the loudspeaker in the room.
2. Microphone x, y, z positions: Specifies the 3D coordinates of the microphone in the room.
3. Room dimensions: Defines the length, width, and height of the room.
4. Loudspeaker directions (θ , ϕ): Indicates the orientation angles of the loudspeaker in spherical coordinates.
5. Microphone directions (θ , ϕ): Indicates the orientation angles of the microphone

¹<https://github.com/mehmetpekmezci/gtu-rir>

in spherical coordinates.

6. Reverberation time (RT60): A key acoustic parameter representing the time it takes for the sound to decay by 60 dB in the room.
7. RIR Data: Contains 88,200 floating-point values ranging from -1 to 1, representing the impulse response waveform captured for each setup.

This dataset provides a rich source of information for analyzing room acoustics and testing RIR generation algorithms.

The GTU-RIR dataset statistics are summarized in Table 3.4, alongside those of the BUT ReverbDB dataset for comparison. Below are the key points regarding the data characteristics:

1. Volume and Surface Values:

- Volume values (in m^3) and surface area values (in m^2) describe the size and enclosure characteristics of the rooms.
- GTU-RIR rooms have half the average volume of BUT ReverbDB, with similar average surface areas, reflecting differences in room geometries.

2. RT60 and SPK-MIC Distance:

- RT60 values are the statistical mode of existing values in a room and range between 0.5 to 2.0 seconds, as calculated using Schroeder's integration method [148].
- The SPK-MIC Distance, representing the three-dimensional Euclidean distance between the loudspeaker and microphone, ranges from 1.3 to 9.5 meters, depending on room dimensions.

3. Validation of RT60 Values:

- The proportional relationship between RT60 and room volume, as dictated by Sabine's formula [34], validates the RT60 calculations.
- The halving of the average RT60 in GTU-RIR compared to BUT ReverbDB aligns with the halving of average room volumes in the GTU-RIR dataset, while surface area averages remain comparable.

This analysis supports the accuracy of the GTU-RIR dataset's RT60 values and highlights the diversity in room acoustics captured within the dataset.

3.3. Microphone Quality

The quality of the microphones was analyzed using the recorded data corresponding to each individual microphone. According to their technical specifications, all microphones

are omni-directional and capable of capturing frequencies in the range of 20 Hz to 20 kHz.

Due to the rotating nature of the system, microphone positions may shift after rotation. For example, microphone 1 may end up in the position of microphone 4 after a 180-degree rotation. Similarly, microphone 2 may align with the position of microphone 5, and microphone 3 with that of microphone 6. The initial positions of these microphones can be seen in Figure 3.27. Note that the numbering in Figure 3.27 starts from 0; for consistency with the explanations in this section, we add 1 to each number—so microphone 1 in the text corresponds to microphone 0 in the figure.

Since we transmit an ESS signal between 20 Hz and 22 kHz, analyzing the frequency spectrum of the recorded sound provides insights into the frequency response of the microphones. As illustrated in the frequency spectrum plots in Figure 3.31 through Figure 3.36, the frequency responses of the microphones are largely similar up to 15 kHz. Beyond this point—typically not perceivable unless one has musical training—minor differences begin to appear. It is important to note that the recordings were made at a sampling rate of 44,100 Hz, which allows capturing frequencies up to 22,050 Hz, in accordance with the Nyquist-Shannon sampling theorem [149]. However, during the evaluation of the generated RIRs, we resample the recorded data to 16,000 Hz, effectively limiting our analysis to frequencies up to 8 kHz.

We also compared all RIR data recorded by microphones 1 to 6 in the GTU-RIR dataset to investigate whether the microphones exhibit different recording behaviors. For this comparison, we selected RIRs extracted from signals transmitted by speakers positioned close to the microphones in question. Specifically, we considered the speaker and microphone positions to be similar if the difference in each coordinate between them was less than 8 centimeters. As shown in Table 3.5, the mean error values across various metrics are consistent for the microphone pairs 1–4, 2–5, and 3–6. This consistency indicates similar acoustic behavior among the microphones. Notably, these pairs are symmetrically arranged, as seen in Figure 3.27, and occupy approximately the same coordinates when the microphone stand is rotated 180 degrees.

Table 3.5: RIR Comparison per microphone.

Microphone Numbers	MEAN MSE	MEAN SSIM	MEAN GLITCH
1 and 4	0.0502	0.2662	118.2379
2 and 5	0.0540	0.2537	120.0975
3 and 6	0.0494	0.2708	123.3362

As illustrated in Figure 3.37, Figure 3.38, Figure 3.39, Figure 3.40, Figure 3.41, and

Figure 3.42, the RIRs extracted from the signals recorded by microphones 1 to 6 exhibit similar shapes, even in cases with the highest MSE, highest glitch count, or lowest SSIM.

In summary, based on the frequency spectrums and RIR comparisons of recordings from microphones 1 to 6, we can conclude that these microphones exhibit similar sound behavior when used in the same room.

3.4. Summary

This chapter introduces the GTU-RIR dataset, a newly developed Room Impulse Response (RIR) dataset. It begins with an overview of publicly available real RIR datasets, highlighting the number of samples they contain and their respective licensing details.

Next, it provides theoretical and practical insights into the RIR measurement process. This includes the data collection algorithm, the frequency ranges of transmitted signals, and key aspects of the mechanical and electrical construction of the system. Essential information such as the bill of materials, connection diagrams, and mechanical drawings are also covered.

The chapter then details the properties of the collected data, including the spatial distribution of loudspeaker and microphone locations, visualized through heatmaps, and a table of data statistics. This table includes metrics such as reverberation time (RT60), samples collected per room, and the physical characteristics of the rooms.

Finally, the chapter explains the structure and content of the collected dataset, providing a comprehensive understanding of its fields and features.



Figure 3.30: Rooms where audio data is collected.

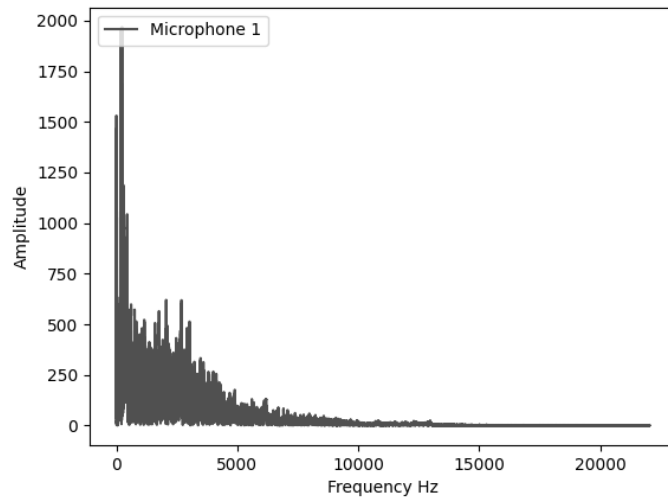


Figure 3.31: Frequency spectrum of microphone 1.

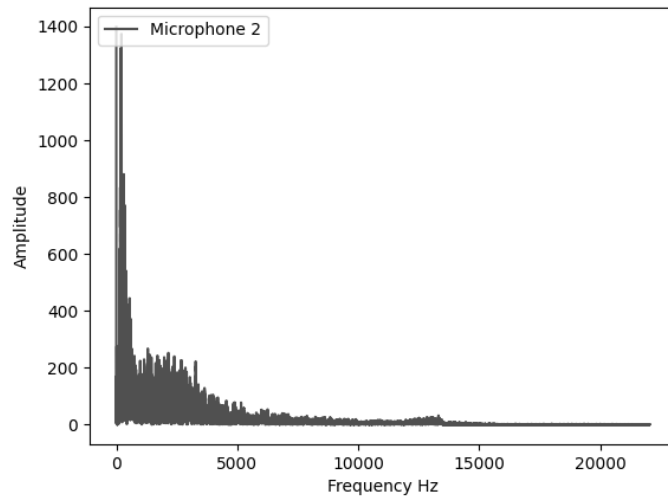


Figure 3.32: Frequency spectrum of microphone 2.

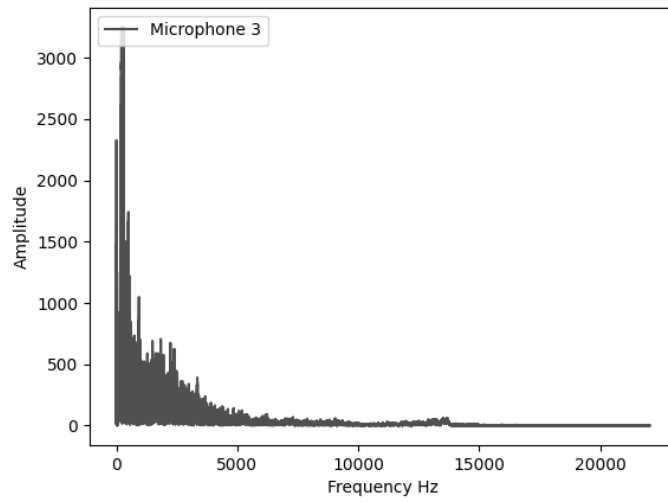


Figure 3.33: Frequency spectrum of microphone 3.

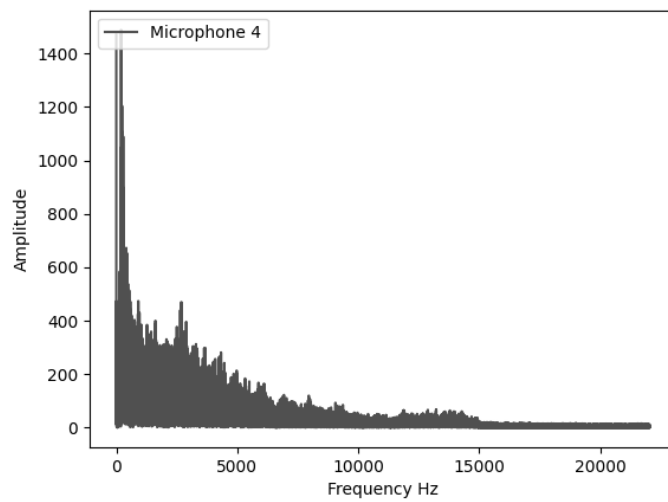


Figure 3.34: Frequency spectrum of microphone 4.

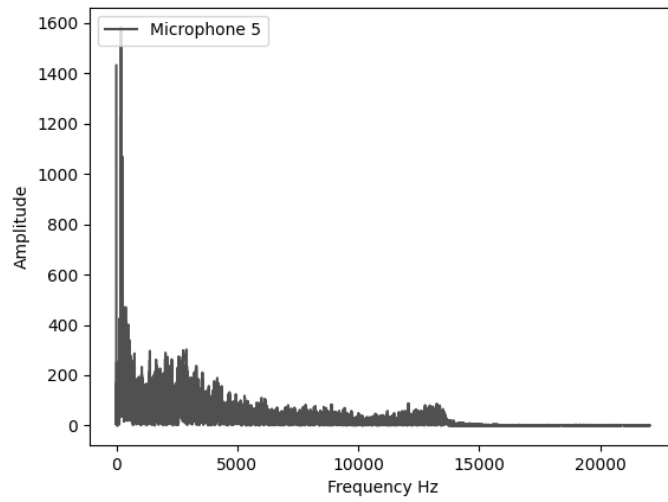


Figure 3.35: Frequency spectrum of microphone 5.

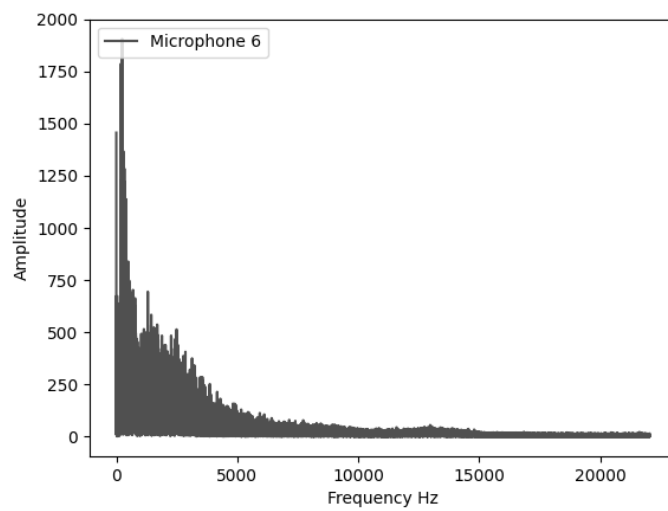


Figure 3.36: Frequency spectrum of microphone 6.

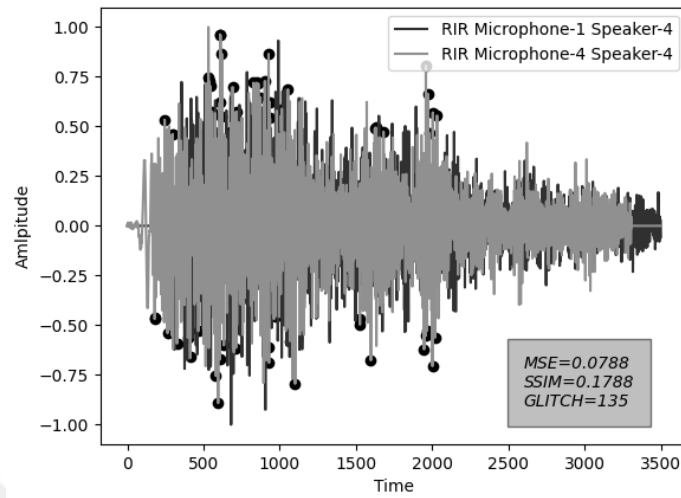


Figure 3.37: The coherence plot of the signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where MSE difference of the records is maximum (Max MSE).

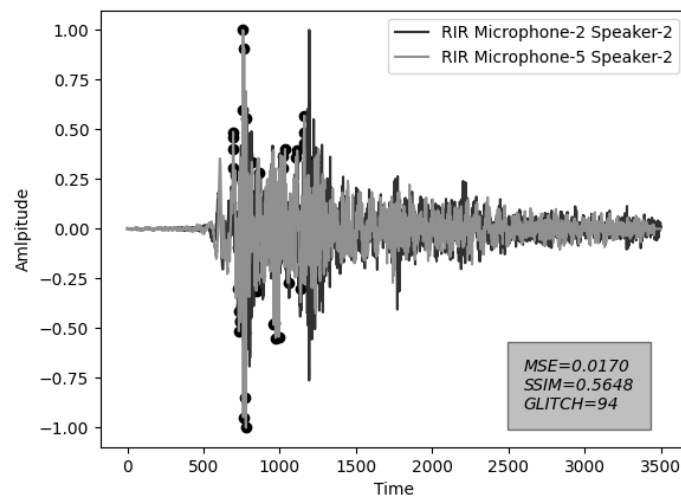


Figure 3.38: The coherence plot of the signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where MSE difference of the records is minimum (Min MSE).

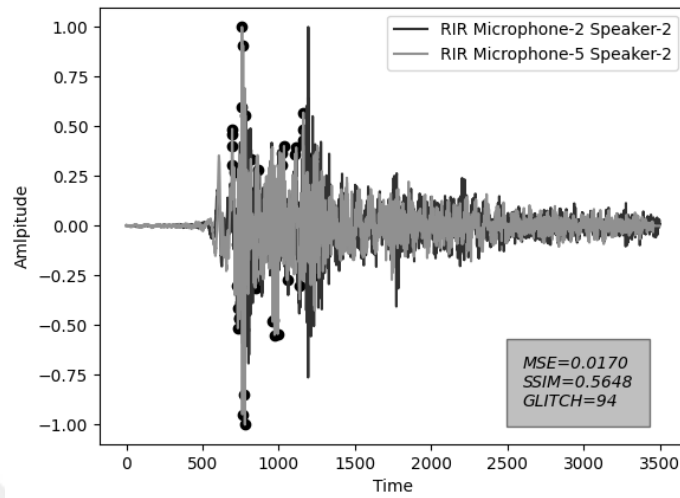


Figure 3.39: The Coherence Plot of the Signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where SSIM difference of the records is maximum (MaxSSIM).

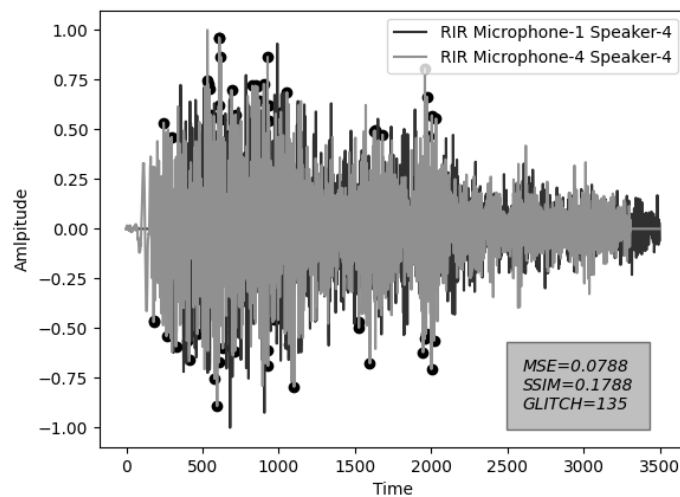


Figure 3.40: The Coherence Plot of the Signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where SSIM difference of the records is minimum (MinSSIM).

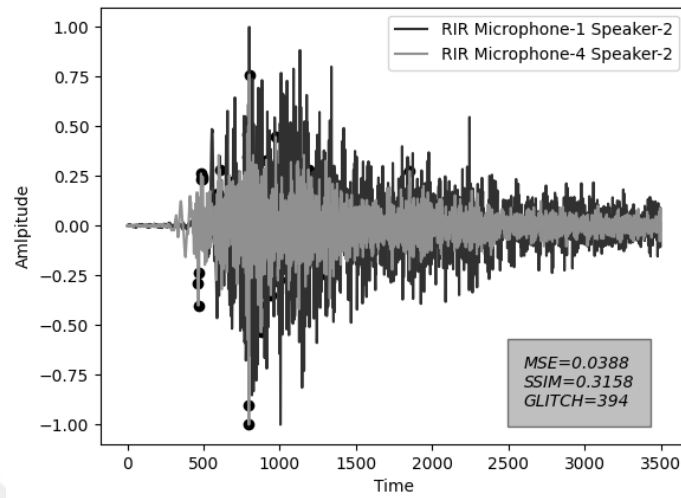


Figure 3.41: The Coherence Plot of the Signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where Glitch Count of the records is maximum (MaxGlitch).

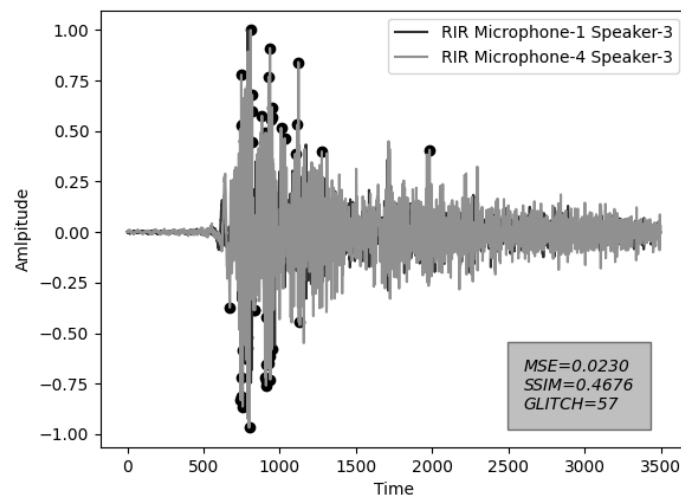


Figure 3.42: The Coherence Plot of the Signals that are recorded in the same room and by the microphones/speakers indicated in the legend of the Figure. This is the record where Glitch Count of the records is minimum (MinGlitch).

4. EVALUATION OF SSIM LOSS FUNCTION

The FAST-RIR GAN model is based on the architecture of the first stage of StackGAN [150]. While StackGAN was originally designed to generate realistic images from textual descriptions, FAST-RIR GAN adapts this approach for generating realistic Room Impulse Response (RIR) signals. Instead of textual inputs, FAST-RIR GAN uses room-specific parameters, including room dimensions, reverberation time (RT60) [55], and the coordinates of the microphone and loudspeaker, to produce RIR signals.

On the other hand, the MESH2IR GAN model [24] utilizes a different approach. Instead of relying on room dimensions and RT60 values, it uses a 3D room mesh as input. To transform the room mesh into a latent vector representation, the MESH2IR GAN incorporates a three-layer Graph Convolutional Network (GCN). This latent vector, combined with microphone and loudspeaker coordinates, forms the input to the StackGAN framework. By leveraging the room's geometric mesh structure, the MESH2IR GAN is capable of generating RIR signals that reflect the spatial and structural intricacies of the environment.

4.1. RIR Generation Using StackGAN with SSIM

The FAST-RIR and MESH2IR GAN models were originally trained using the Mean Squared Error (MSE) loss function. To explore the potential benefits of using the Structural Similarity (SSIM) loss function, both models were modified by replacing the MSE loss with SSIM. The SSIM loss function was applied in two ways: directly on the RIR data and on the Mel-Frequency Cepstrum Coefficients (MFCC) extracted from the RIR data.

The SSIM loss functions used are as follows:

- Direct SSIM loss on RIR data as shown in Equation 4.1
- SSIM loss on MFCC-transformed RIR data as shown in Equation 4.2

$$L_1 = 1 - \text{ssim}(\text{RIR}_{\text{real}}, \text{RIR}_{\text{fake}}) \quad (4.1)$$

$$L_2 = 1 - \text{ssim}(\text{mfcc}(\text{RIR}_{\text{real}}), \text{mfcc}(\text{RIR}_{\text{fake}})) \quad (4.2)$$

After implementing these changes, the FAST-RIR GAN model was re-trained using synthetic data generated by the Diffuse Acoustic Simulator (DAS) method, as specified in the original FAST-RIR paper. Similarly, the MESH2IR GAN model was re-trained using the synthetic GWA dataset, as outlined in the MESH2IR paper.

These modifications allowed for an evaluation of the SSIM-based loss functions in generating RIR signals with improved perceptual similarity and better acoustic fidelity compared to the MSE-based approach.

Algorithm 2 Glitch Finding Algorithm

```

1: function Find_Glitch_Points(generated_rir_signal_array, real_rir_signal_array)
2:   insensitivity  $\leftarrow$  3
3:   glitch_threshold  $\leftarrow$  standard_deviation(real_rir_signal_array) *
      insensitivity
4:   glitch_points  $\leftarrow$  empty_list()
5:   for  $i = 0$  to len(generated_rir_signal_array) do
6:     if  $abs(abs(generated\_rir\_signal\_array[i]) -$ 
       $abs(real\_rir\_signal\_array[i])) > glitch\_threshold$  then
7:       glitch_points.append( $i$ )
8:     end if
9:   end for
10:  return glitch_points
11: end function

```

To assess the performance of the trained GAN models, we used room geometries and loudspeaker/microphone positions from the GTU-RIR dataset to generate RIRs for those configurations. The generated RIRs were then compared to the real RIRs using both MSE and SSIM metrics. To ensure the evaluation’s robustness and data independence, we repeated the same process with another real dataset, BUT ReverbDB.

Additionally, we employed a glitch finder algorithm to detect anomalies in the generated RIR signals as shown in Algorithm 2. This algorithm identifies glitch points by analyzing the difference between the generated and real signals, flagging points where the difference surpasses the standard deviation of the real signal. The identified glitch points are visually represented as black spots in Figure 2.14, illustrating the nature and distribution of the detected glitches.

4.2. Experiments and Evaluation

The comparison process between real and generated (fake) signal data is illustrated in Figure 4.1. In this process, the FAST-RIR GAN model generates an RIR signal using inputs such as room dimensions, loudspeaker and microphone coordinates, and the

room’s reverberation time (RT60). Similarly, the MESH2IR model generates an RIR signal by utilizing the loudspeaker and microphone coordinates along with the room’s mesh geometry as inputs.

The FAST-RIR GAN model, as shown in Figure 4.1, features a generator with five layers of 1D transposed convolution and a discriminator with four CNN layers. The generator network takes a ten-element input vector containing the microphone and loudspeaker coordinates, room dimensions, and RT60 value, and produces an RIR signal represented as a list of 4096 float values. The generator network includes 115 million trainable parameters, while the discriminator network has 15 million trainable parameters.

The MESH2IR GAN model, also depicted in Figure 4.1, integrates a three-layer Graph Convolutional Network (GCN) called MeshNet, which processes the room mesh. To simplify the computational load, the room mesh is down-sampled to 2000 nodes before being passed to the MeshNet, which contains 3560 trainable parameters. The remaining parts of the MESH2IR network, including the generator and discriminator, match the parameter counts of the FAST-RIR model. This model takes microphone and loudspeaker coordinates along with the room mesh as inputs.

Both GAN models, inspired by StacGAN-I [150], are presented together in the figure because their architectures share similarities, particularly in their generative and discriminative components. However, the MESH2IR model’s inclusion of the MeshNet for processing room geometry distinguishes it from FAST-RIR.

As illustrated in the ”post-processing” section of Figure 4.1, prior to comparing real and generated (fake) RIRs, we perform two key adjustments: aligning the maximum points of the waves along the x-axis and equalizing the maximum points of the waves along the y-axis to 1, as depicted in Figure 4.2.

During experiments, we observed that power equalization—scaling the RIR waveforms based on their maximum values—helps focus on how well the generated RIR matches the real RIR by removing the influence of power loss. While power loss can still be calculated afterward, our primary aim here is to smooth out glitches in the generated RIRs for more accurate comparisons.

To evaluate the quality of the alignment, both Mean Squared Error (MSE) and Structural Similarity Index Measure (SSIM) distances are computed between the real and generated RIRs. This dual metric approach allows us to assess improvements across both measures.

We applied a coordinate transformation of (X, Y, Z) to $(X, WIDTH - Y, Z)$ before feeding the coordinate inputs to the neural network, as illustrated in Figure 4.3. This transformation significantly improved the generated RIR signal quality, resulting in lower MSE and higher SSIM values. These results suggest that GAN architectures are sensitive to the choice of coordinate systems and we should be sure of using the

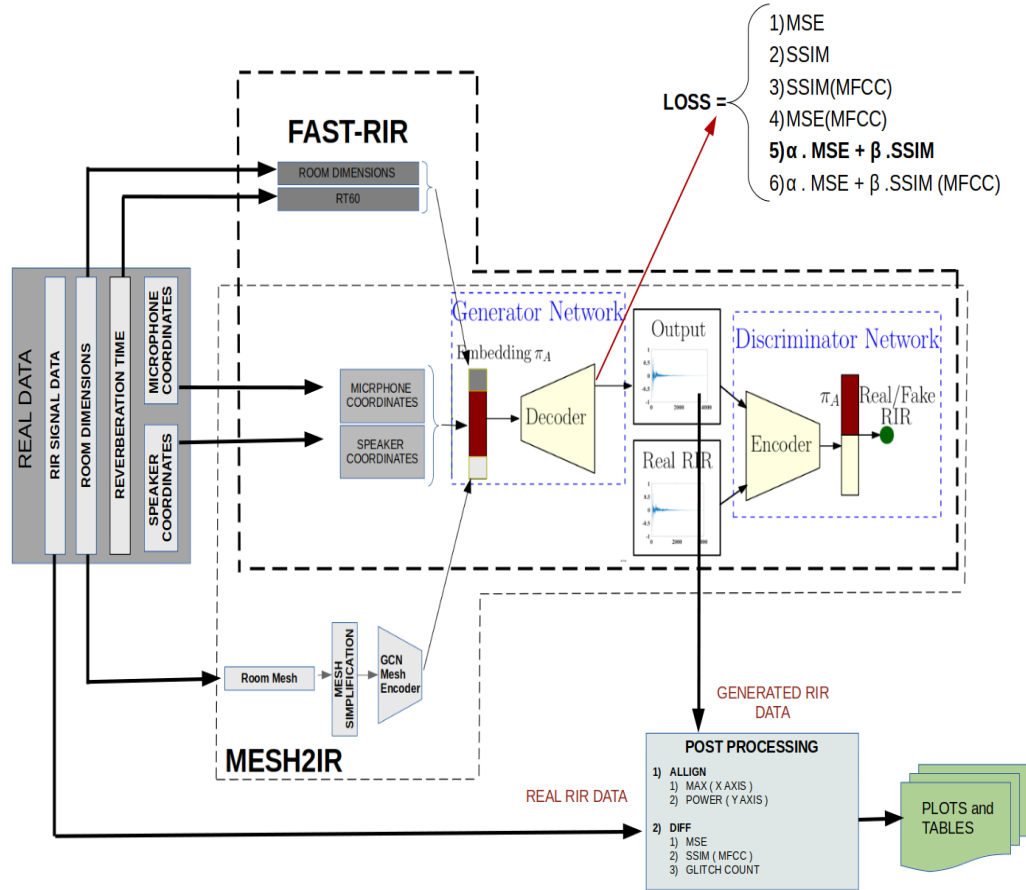


Figure 4.1: Comparing FAST-RIR [23] and MESH2IR [24] generated RIRs with real RIRs. The architecture outlined with a dark dashed line represents FAST-RIR, while the architecture outlined with a light dashed line represents MESH2IR.

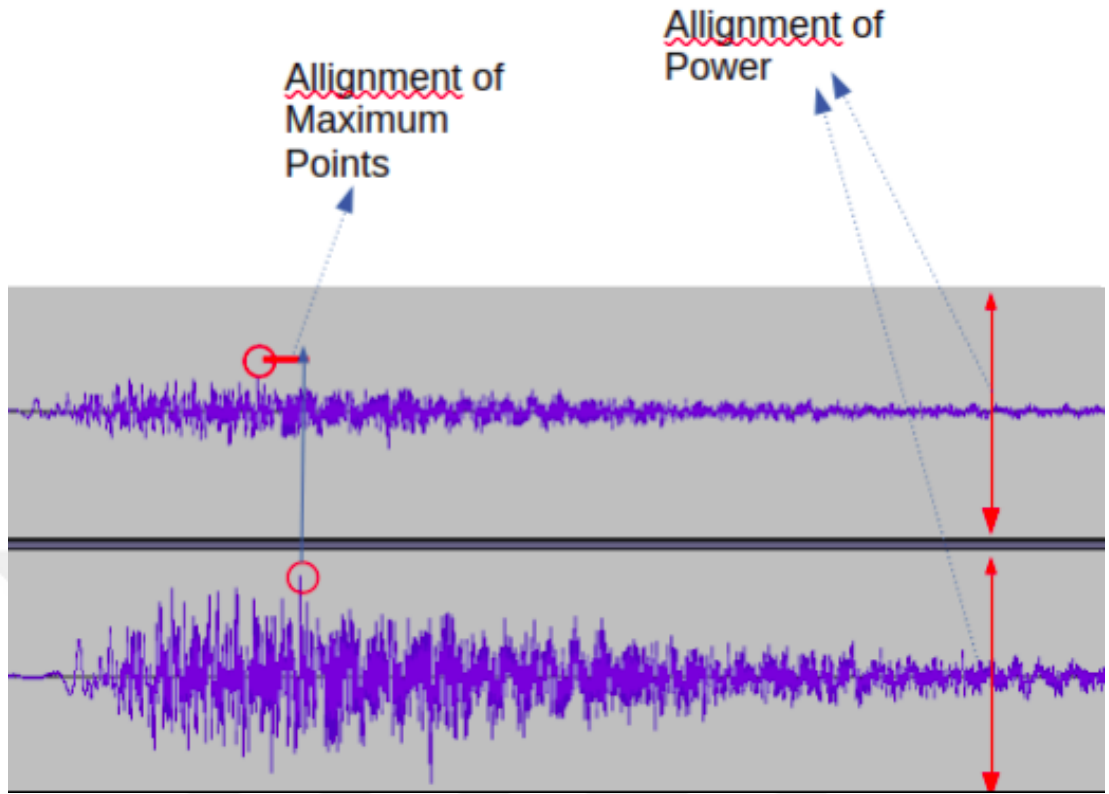


Figure 4.2: Alignment of waves.

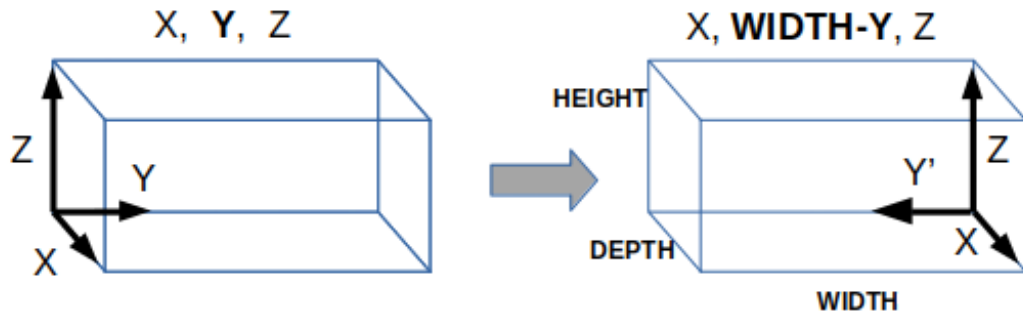


Figure 4.3: Coordinate systems conversion.

same coordinte system for training and test data. Additionally, increasing the number of MFCC coefficients from 40 to 4096 enhanced the SSIM values by over 50%.

A total of 24 experiments were conducted using the FAST-RIR and MESH2IR GAN models, trained with different loss functions, MSE, SSIM, and MFCC-based SSIM, on the GTU-RIR and BUT ReverbDB datasets. The generated RIRs were compared with

the real RIRs using the MSE, SSIM, and Glitch Count metrics, as detailed in Table 5.1.

Each experimental setup followed these steps:

1. Selection of the GAN model (FAST-RIR or MESH2IR).
2. Replacement of the generator's MSE loss function with SSIM, MFCC-based SSIM, or a combination of these loss functions.
3. Training of the model using synthetic data from the respective research papers.
4. Selection of a real dataset (GTU-RIR or BUT ReverbDB) for evaluation.
5. Generation of RIRs based on the dataset metadata using the trained GAN model.
6. Comparison of generated and real RIRs using MSE, SSIM, and Glitch Count metrics.
7. Generation of detailed reports and plots summarizing the findings.

The results in Table 5.1 show that GAN models trained with a combined MSE + SSIM loss function produced the best results. These models consistently yielded the lowest Glitch Count and MSE values and the highest SSIM values across both datasets (GTU-RIR and BUT ReverbDB) and models (FAST-RIR and MESH2IR).

Additionally, MSE values for the GTU-RIR dataset were higher compared to BUT ReverbDB, which is attributed to the greater spatial distribution of the GTU-RIR data, as evidenced by the spatial heatmaps in Figures 3.1 and 3.2.

We generated coherence plots to visually assess the reduction in glitches when using GAN models trained with the SSIM + MSE loss function. Each group of coherence plots includes six subplots, showing the RIR signals generated by two different GAN models—FAST-RIR and MESH2IR—trained with three distinct loss functions: MSE, SSIM, and SSIM + MSE.

These plots enable a comparative analysis of the generated RIR signals, highlighting the improvement in signal smoothness and glitch reduction achieved by incorporating the SSIM component into the loss function. The visualized coherence underscores the effectiveness of combining SSIM and MSE in training GAN models for RIR generation.

In Figure 4.4, we selected an RIR from the GTU-RIR dataset with an average glitch count between the generated and real RIRs. This same real RIR is used consistently across all six subplots in the figure. The results demonstrate noticeable improvements when the GAN models are trained using SSIM and SSIM+MSE loss functions.

The plots clearly show that both FAST-RIR and MESH2IR models generate RIRs with fewer glitches when their respective GANs are trained with the SSIM+MSE loss function. Similar experiments conducted using the BUT ReverbDB dataset confirm these findings, as illustrated in Figure 4.5.

Table 4.1: Comparison of loss values based on SSIM and MSE. Combined loss improves the results (in bold) over MSE or SSIM alone while reducing glitch counts in both datasets.

GAN Model	Training Loss Fn.	MEAN MSE	MEAN SSIM	MEAN GLITCH	
GTU-RIR	FASTRIR	MSE(MFCC)	0.031	0.354	133
		SSIM(MFCC)	0.038	0.295	103
		SSIM(MFCC) +MSE	0.041	0.225	136
		MSE	0.048	0.214	155
		SSIM	0.038	0.306	114
		SSIM+MSE	0.040	0.318	113
	MESH2IR	MSE(MFCC)	0.052	0.232	161
		SSIM(MFCC)	0.058	0.154	217
		SSIM(MFCC) +MSE	0.04	0.313	139
		MSE	0.043	0.299	130
		SSIM	0.051	0.270	167
		SSIM+MSE	0.033	0.361	132
BUT ReverbDB	FASTRIR	MSE(MFCC)	0.021	0.617	136
		SSIM(MFCC)	0.029	0.256	307
		SSIM(MFCC) +MSE	0.022	0.500	138
		MSE	0.015	0.562	116
		SSIM	0.072	0.374	438
		SSIM+MSE	0.015	0.623	93
	MESH2IR	MSE(MFCC)	0.042	0.367	378
		SSIM(MFCC)	0.043	0.211	570
		SSIM(MFCC) +MSE	0.029	0.497	217
		MSE	0.028	0.494	241
		SSIM	0.037	0.414	336
		SSIM+MSE	0.023	0.567	161

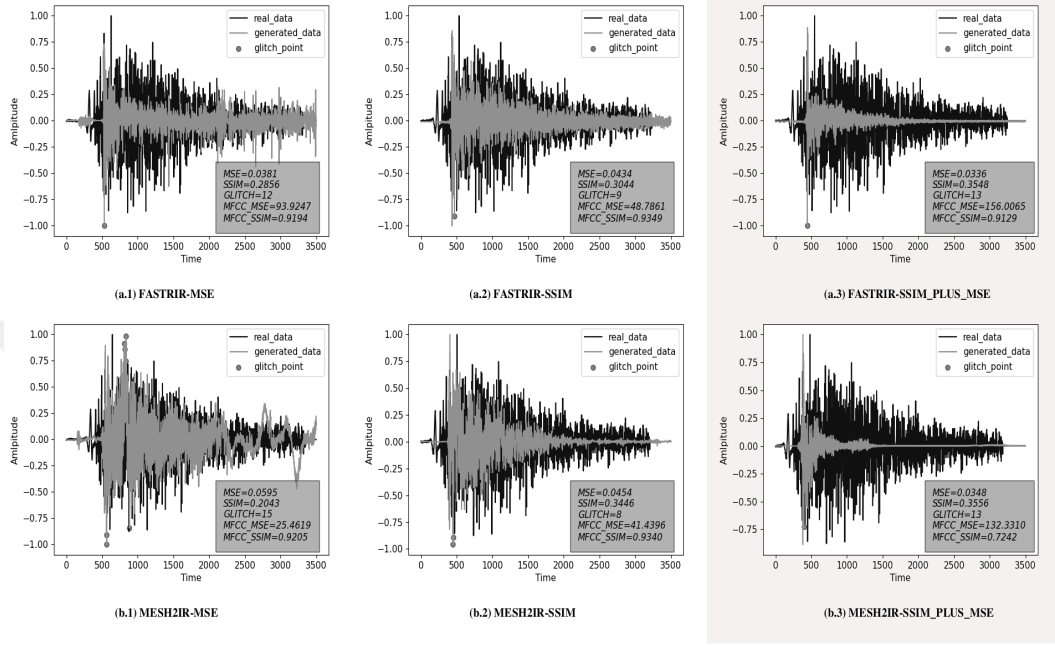


Figure 4.4: The coherence plots illustrate the comparison between real and generated room impulse responses (RIRs) for two GAN models each trained with three different loss functions: (a.1) FAST-RIR trained with MSE, (a.2) FAST-RIR trained with SSIM, (a.3) FAST-RIR trained with SSIM+MSE, (b.1) MESH2IR trained with MSE, (b.2) MESH2IR trained with SSIM, and (b.3) MESH2IR trained with SSIM+MSE. Specifically, a specific data point is selected from the **GTU-RIR** dataset, and the real RIR for that point is compared to the RIRs generated by each of the four GAN models. We can see that the GAN models trained with the SSIM loss function produce RIRs with fewer glitches or generate RIRs that better match the real RIRs.

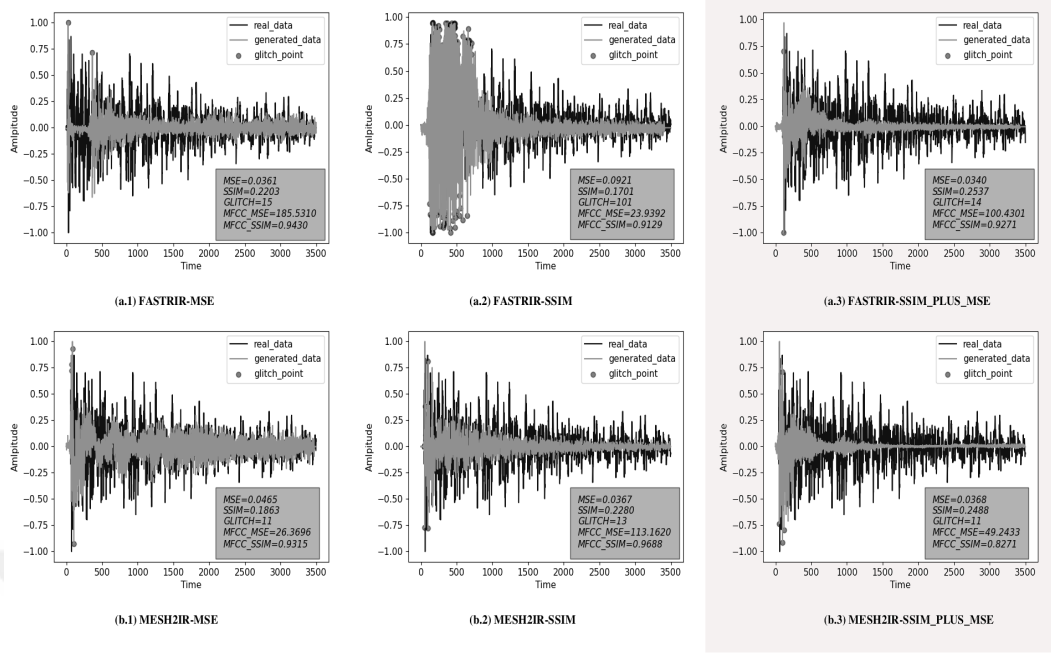


Figure 4.5: The coherence plots for the **BUT ReverbDB** dataset. We show the same measurements as in Figure 4.4.

Training the GANs with the SSIM+MSE loss function benefits from the strength of SSIM in identifying perceptual differences between real and generated RIRs. Consequently, the generated RIRs align more closely with the desired characteristics of real RIRs, significantly reducing glitches in the generated signals. These improvements are consistently observed in both the GTU-RIR and BUT ReverbDB datasets, as depicted in Figures 4.4 and 4.5.

Combining MSE and SSIM loss functions yields better results due to the complementary properties of these metrics. The MSE loss function calculates the cumulative differences across all data points, treating all deviations equally. This approach means that large relative differences between small values and small relative differences between large values contribute equally to the MSE value. Consequently, MSE alone may fail to distinguish subtle variations in small signal values, as demonstrated in (a.1) of Figure 4.4.

On the other hand, SSIM is designed to assess local structural differences, making it more sensitive to variations in texture and patterns. This sensitivity allows SSIM to better handle discrepancies in small signal values where MSE might fall short. However, SSIM can sometimes struggle to capture global variations or subtle differences when

those differences are not localized enough, as shown in (a.2) of Figure 4.5.

By combining MSE and SSIM loss functions, the model benefits from their respective strengths: MSE ensures global consistency, while SSIM focuses on preserving local structural details. This combination effectively reduces spikes and enhances the alignment between real and generated signals.

4.2.1. A Human-Centric Evaluation of Generated RIRs

To further assess the quality of the generated RIRs, we conducted a listening test where participants were asked to rate the similarity between songs reverberated using synthetic RIRs and their corresponding real recorded reverberated versions, specifically in terms of reverberation characteristics and the presence of crackling artifacts, as summarized in Table ?? . All the audio samples used in the survey, along with the coherence plots, are available for download¹.

The survey was conducted with 11 participants, each evaluating a different randomly selected sample from the GTU-RIR dataset. Each sample corresponded to a recording made in a distinct room within the Computer Engineering Department of GTU. The participants represented a diverse range of profiles in terms of age (ranging from 20 to 55), academic background (bachelor's, master's, and Ph.D. levels), profession (including lawyer, computer engineer, electrical engineer, mechanical engineer, electronics engineer, and telecommunications engineer), and musical education level.

For each room, participants were presented with the original song recorded using the selected microphone in that specific environment. Additionally, they were provided with the same song reverberated using both the real RIR and the synthetic RIRs generated by the models. Participants were then asked to rate the similarity between the real recorded song and the artificially reverberated versions (produced using both real and synthetic RIRs) on a scale from 1 to 10, where 1 represents the lowest similarity and 10 represents the highest similarity.

According to the survey results presented in Table 4.2, the MESHTAE model achieved the second highest similarity scores (MEAN Human Rating Score), following the real RIR similarity scores. This result highlights MESHTAE as the most successful RIR generation model among those evaluated in this study. Furthermore, the SSIM metric values illustrated in the coherence plots align well with human perception, reinforcing the reliability of SSIM in capturing perceptual similarity. Although MESHTAE is discussed in detail in the next chapter (Chapter 5), we included this model in the survey to specifically obtain human perception scores for comparison within this evaluation.

¹HUMAN_EVALUATION_SURVEY_QUESTIONNAIRE

Table 4.2: RIR comparison metrics and Mean Human Rating Scores of Survey on Human Perception Quality.

Model	MEAN MSE	MEAN SSIM	MEAN GLITCH	MEAN Human Rating Score
REAL RIR	0.000	1.000	0	8.36/10
FASTRIR-MSE	0.048	0.214	155	6.81/10
FASTRIR-SSIM	0.038	0.306	114	7.54/10
FASTRIR-SSIM_PLUS_MSE	0.040	0.318	113	7.81/10
MESH2IR-MSE	0.043	0.299	130	7.45/10
MESH2IR-SSIM	0.051	0.270	167	7.36/10
MESH2IR-SSIM_PLUS_MSE	0.033	0.361	132	7.90/10
MESHTAE	0.032	0.363	130	8.18/10

4.3. Summary

This chapter evaluates the effectiveness of the Structural Similarity (SSIM) loss function in reducing glitches within generated room impulse response (RIR) signals produced by generative adversarial networks (GANs). To address the issue of glitches, we incorporated the SSIM loss function into the training process of GAN models. Using this approach, the GANs trained with the SSIM loss function demonstrated a significant reduction in glitches and MSE and increase in SSIM metrics. This chapter also presents comprehensive comparison tables and plots, illustrating the performance of GAN models trained with MSE, SSIM, and a combination of these loss functions.

5. GEOMETRIC EMBEDDING IMPROVEMENT

Room Impulse Responses (RIRs) can be generated using a variety of neural network architectures [23–32, 58]. Among these, only MESH2IR [24] and Listen2Scene [31] incorporate room geometry as an explicit input to the network. These models integrate geometric embedding mechanisms with GAN-based RIR generators. However, a key limitation of these architectures is the requirement to retrain both the geometric and acoustic components for each new room geometry, limiting generalizability and scalability.

To overcome this constraint, we propose two alternative models based on distinct methodologies. In both approaches, we explicitly decouple the geometric and acoustic components of the RIR generation pipeline, enabling modular training and reuse of geometric embeddings across different acoustic scenarios.

In the first approach, we developed a Mesh Transformer Auto-Encoder model, illustrated in Figure 5.2, inspired by the MeshMAE architecture [118]. This model was trained on a synthetically generated dataset of 3D apartment meshes created using the Trimesh library¹. The dataset includes a wide variety of room configurations and spatial arrangements, generated via a custom apartment mesh generator script², designed to simulate diverse architectural layouts for effective generalization.

We extended the MESH2IR model by substituting its geometric embedding module with the pre-trained Mesh Transformer Auto-Encoder. This second model is named MESHTAE. To preserve the learned geometric representations, we adapted the training procedure such that the parameters of the Mesh Transformer Auto-Encoder remain frozen during the training of the acoustic model (RIR generator). Experimental results demonstrate that it is indeed possible to decouple the training of geometric and acoustic components and effectively utilize pre-trained geometric embeddings within RIR generation networks. This modular design removes the necessity of retraining the geometric model for each new room configuration, thereby improving scalability and computational efficiency.

In the second approach, we projected the 3D apartment mesh onto a 2D plane to generate the corresponding floor plan. Sound transmitter and receiver points were then placed on this floor plan, and 2D ray-casting images were generated from each of these points. In these images, red regions represent first-order reflections, while blue regions indicate second-order reflections, as illustrated in Figure 5.1. The overall ray-casting process is

¹Trimesh.org

²Apartment Mesh Generator Script

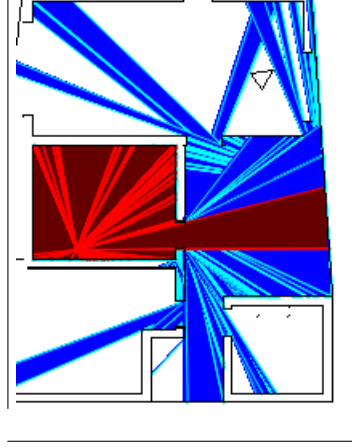


Figure 5.1: Example ray-casted floor plan image from the MESH2IR training dataset:

This image illustrates a 2D floor plan of an apartment flat, generated from a 3D mesh projection, with ray casting applied to simulate acoustic paths used during MESH2IR training.

depicted in Figure 5.8.

To encode the geometric information, we employed the pre-trained Stable Diffusion Variational Encoder by Kouzelis [151], extracting a latent vector from each ray-casting image. These latent vectors—corresponding to the microphone and loudspeaker positions—were then concatenated and used as the geometric input to the acoustic GAN model. This second model is referred to as RCRIR. Notably, this approach eliminates the need to train a separate geometric embedding model alongside the acoustic model.

However, experimental results show that the first approach—based on the Mesh Transformer Auto-Encoder—achieves superior performance compared to this ray-casting-based method.

For comparative purposes, we also trained an acoustic model using only the room or apartment flat dimensions as geometric input. We named this model as DIMRIR. This minimal geometric representation resulted in the weakest performance among all approaches, thereby confirming that enhancing the geometric modeling component significantly improves the quality of the generated RIRs. This observation reinforces the effectiveness of the more sophisticated geometric encoding strategies employed in the first two approaches.

5.1. Separating Geometric and Acoustic Models

The MESH2IR [24] RIR generation model processes room meshes by converting them into latent vector representations using a three-layer Graph Convolutional Network

MESH TRANSFORMER AUTO-ENCODER

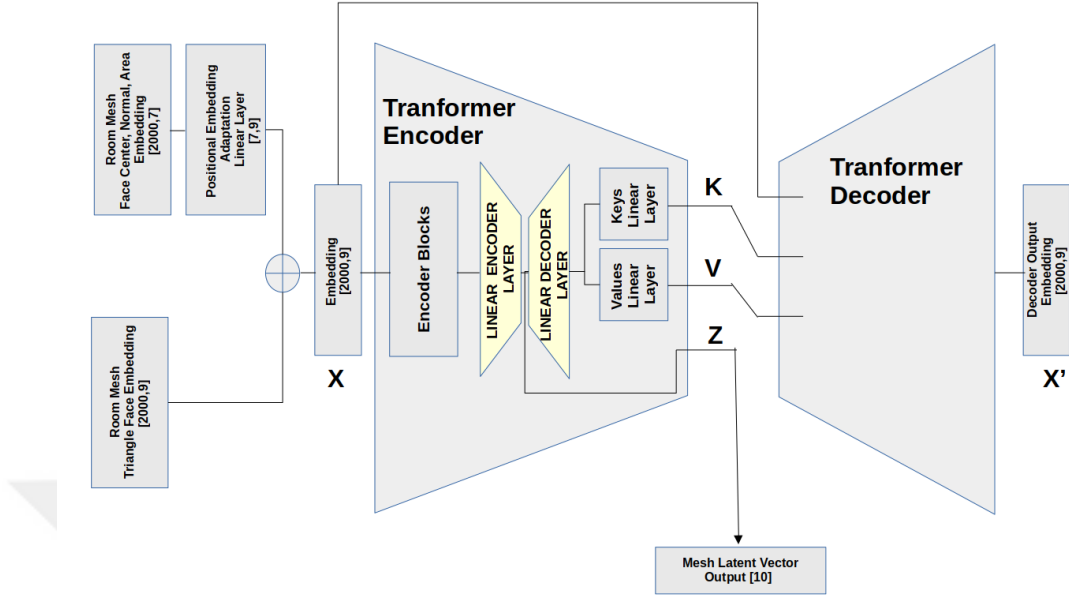


Figure 5.2: Mesh transformer auto-encoder architecture.

(GCN), which serves as the geometric model. The resulting latent vector is concatenated with microphone and loudspeaker coordinates to form the input for the StackGAN model, which serves as the acoustic model.

In the MESH2IR model, the geometric and acoustic models are trained together in a single cycle. During backpropagation, the parameters of the geometric model (the GCN) are updated after the acoustic model (StackGAN), optimizing the geometric model specifically for RIR generation. This tightly coupled training approach necessitates re-training the geometric model for each new type of room geometry, which is time-consuming and inefficient.

To address these limitations, we propose decoupling the geometric and acoustic models and training them separately. By training the geometric model as an auto-encoder, we ensure it can independently generate embeddings that capture the geometric features of the room without requiring re-training in the RIR generator. This decoupled approach allows for pre-trained geometric embeddings to be reused, saving training time and computational resources.

5.1.1. The MESHTAE Model

The proposed model, MESHTAE, is composed of two parts:

1. MESH-TRANSFORMER-AE: A transformer-based mesh auto-encoder that serves as the geometric model.
2. StackGAN: A GAN model for RIR generation that serves as the acoustic model.

As depicted in Figure 5.12, the geometric model in MESHTAE is pre-trained and used without re-training during RIR generation, represented by a unidirectional arrow.

The Mesh Transformer Auto-Encoder (MESH-TRANSFORMER-AE) is a transformer-based architecture designed to process 3D mesh data by encoding and reconstructing mesh structures, as illustrated in Figure 5.2. In addition to mesh reconstruction, the model extracts a latent vector representation that encapsulates the geometric characteristics of the mesh. Our design was inspired by recent advancements in mesh-based neural architectures, including MESHMAE [118], MeshGPT [119], and NeRFMeshing [117]. Among these, MESHMAE was identified as the most aligned with our objectives, and based on its proposed design, we developed the MESH-TRANSFORMER-AE, tailored specifically for the requirements of room acoustic modeling.

Initially, we explored graph neural network (GNN) architectures to construct an auto-encoder capable of simultaneously reconstructing both the connectivity and the coordinates of mesh nodes, as discussed in Chapter 2. However, these models were unable to reconstruct meshes with visually accurate fidelity, leading us to adopt a transformer-based alternative.

A key aspect of our approach is the representation of mesh surfaces as **reflection faces**, which are critical for modeling acoustic reflections. The input embedding to the MESH-TRANSFORMER-AE is thus carefully constructed to encode reflection-related geometric properties. This encoding allows the subsequent RIR-generating GAN to leverage the learned latent vector effectively for accurate impulse response generation.

Each mesh face is represented as a 16-dimensional vector: the first 9 values correspond to the 3D coordinates of its three triangle vertices, and the remaining 7 describe the geometric properties of the face—namely, the centroid coordinates, surface normal vector, and area. To incorporate positional information, the final 7 features are passed through a linear transformation layer that maps them to a 9-dimensional space. This transformed representation is then added to the original vertex coordinates to form the final input embedding.

The resulting embeddings are processed by the encoder module of the transformer auto-encoder, which comprises a single block and 8 attention heads. Unlike MESHMAE, which applies masking strategies to improve training efficiency, we found that our

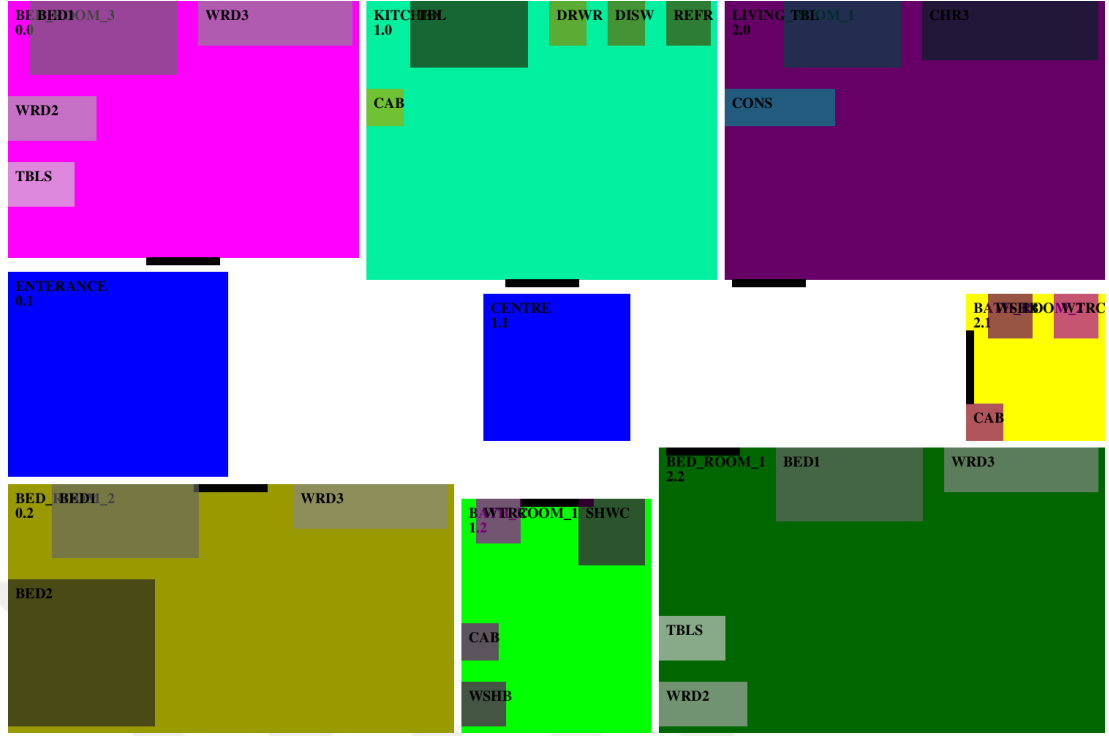


Figure 5.3: Floor plan of a sample in synthetic apartment flat dataset.

model achieved sufficient reconstruction accuracy without the need for masking.

Latent Vector Extraction:

The output of the transformer encoder is typically of the same size as the input. To derive a latent vector, we introduced a two-layer linear auto-encoder between the encoder's output and the transformer's Key/Value layers. This additional module compresses the encoded representation into a 10-dimensional latent vector, denoted as Z in Figure 5.2, which serves as the geometric embedding of the mesh.

RIR Generation:

The latent vector generated by the MESH-TRANSFORMER-AE, along with microphone and loudspeaker coordinates, is passed to the StackGAN (acoustic model) for RIR generation. This modular design allows the geometric and acoustic models to function independently, enabling the use of pre-trained geometric embeddings in various RIR generation tasks.

By adopting this approach, MESHTAE eliminates the need for re-training the geometric model for new room geometry types, offering a more efficient and flexible solution for RIR generation.

We trained the MESH-TRANSFORMER-AE network using a synthetic dataset containing over one million samples. Each sample represents a 3-room apartment flat mesh

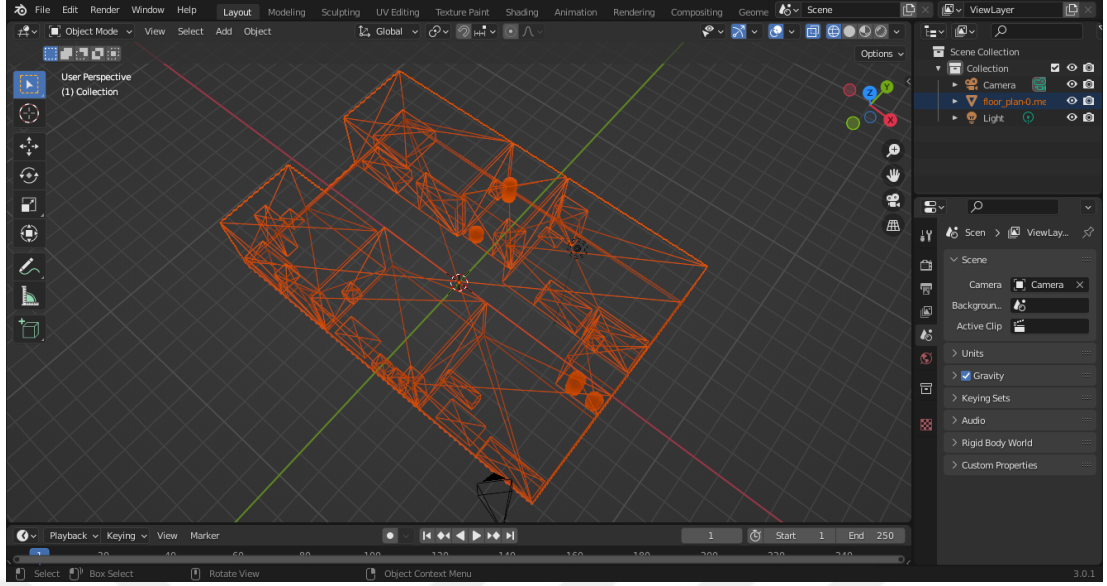


Figure 5.4: 3D mesh view of a sample in synthetic apartment flat dataset.

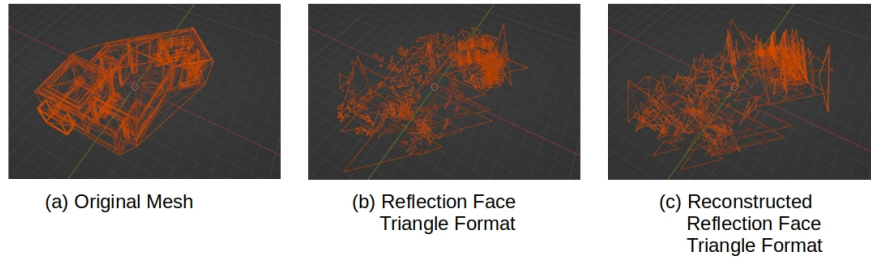


Figure 5.5: Best mesh reconstruction result for a room geometry in the MESH2IR’s RIR dataset.

generated randomly using the apartment mesh generator script. Example outputs of the script are shown in Figures 5.3 and 5.4. The MESH-TRANSFORMER-AE network is solely used to obtain embeddings of the given room geometry and is not further trained during the training of the StackGAN network, which serves as the acoustic component of the RIR generation model.

The mesh transformer autoencoder (MESH-TRANSFORMER-AE) model performance is illustrated in Figures 5.5, 5.6, and 5.7. These are the mesh geometries of MESH2IR’s RIR dataset. The MESH-TRANSFORMER-AE model is not trained using this dataset, MESH-TRANSFORMER-AE is only used to generate latent vector of the room geometries within this dataset.

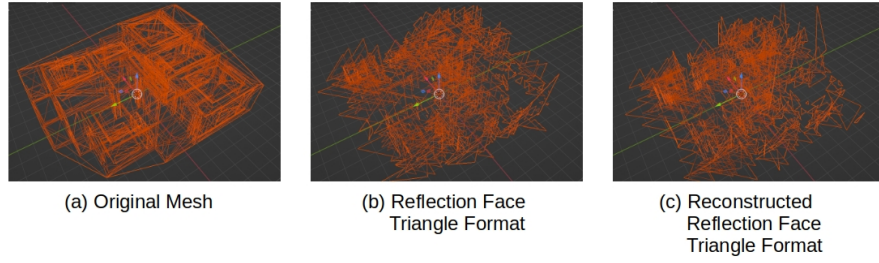


Figure 5.6: Average mesh reconstruction result for a room geometry in the MESH2IR's RIR dataset.

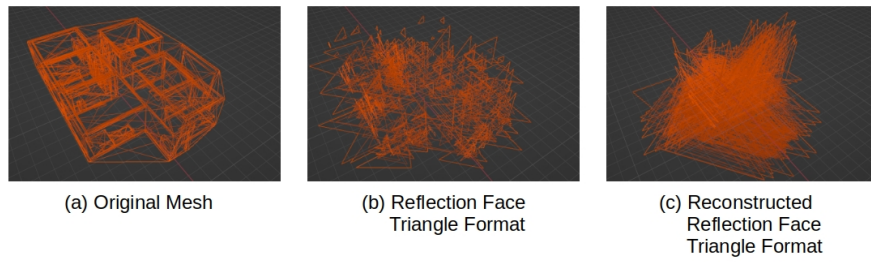


Figure 5.7: Worst mesh reconstruction result for a room geometry in the MESH2IR's RIR dataset.

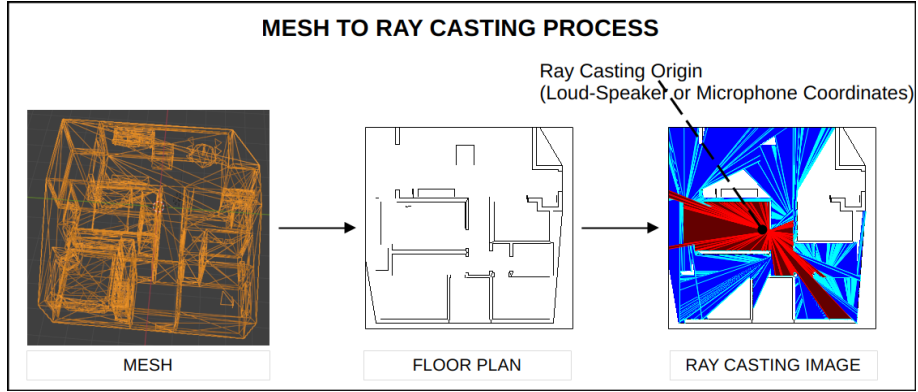


Figure 5.8: Ray casting process: The 3D mesh is initially projected onto a 2D plane using the Trimesh library to generate a floor plan image. Subsequently, two-dimensional ray casting images are produced based on the algorithm described in [153] and rendered using Pygame. These images simulate sound propagation behavior originating from specified source and receiver points within the floor plan layout.

To evaluate the performance of the MESHTAE model, we utilized room geometries and loudspeaker/microphone positions from the GTU-RIR [67] dataset. RIRs were generated for these geometries and positions and compared with the real RIRs using both the mean squared error (MSE) and structural similarity (SSIM) metrics. To ensure data independence, we repeated the same comparison using another real dataset, BUT ReverbDB [152].

5.1.2. The RCRIR Model

As a second approach, we adopted a ray-casting-based model to explore whether incorporating information about reflective surfaces and their acoustic reflection behavior—encoded through color-coded rays and ray-casted regions, as illustrated in Figure 5.8—could enhance the quality of the generated Room Impulse Responses (RIRs).

In this approach, to construct a geometric representation of the environment, we first project the 3D mesh onto a 2D plane to obtain the floor plan of the apartment flat or room. Subsequently, we applied a two-dimensional ray casting algorithm [153] to simulate sound reflections up to the second order. The ray-casted regions were visualized using Pygame, where first-order reflections are depicted in red and second-order reflections in blue. The origins of the second-order rays are determined by the endpoints of the corresponding first-order rays. To capture both the transmission and

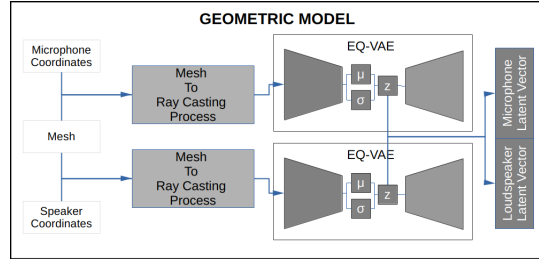


Figure 5.9: RCRIR geometric model: Ray casting images are generated for both the microphone and loudspeaker positions. These images are then processed individually using the pre-trained EQ-VAE model [151] to extract their latent vector representations. The resulting vectors are subsequently concatenated to form the final geometric representation.

reception characteristics of the space, separate ray casting images were generated for the loudspeaker and microphone coordinates, treating each as the respective ray casting origin.

Latent Vector Extraction:

After generating the ray casting images for both the microphone and loudspeaker coordinates, we input each image separately into the pre-trained EQ-VAE model [151] to obtain their respective latent vector representations. As illustrated in Figure 5.9, the resulting latent vectors are concatenated to form a unified geometric representation. This combined vector is then used as input to the acoustic GAN model for RIR generation.

RIR Generation:

The generated latent vector, together with the microphone and loudspeaker coordinates, is provided as input to the StackGAN-based acoustic model for RIR generation, as illustrated in Figure 5.10. Similar to the MESHTAE-based approach, the RCRIR model architecture decouples the geometric and acoustic components. This modular design allows the acoustic model to operate with pre-extracted geometric embeddings, thereby eliminating the need to retrain the geometric model for each new room geometry.

The RCRIR model does not require training a geometric module, as the ray casting process is a deterministic, non-neural operation, and the EQ-VAE used for extracting latent image representations is a pre-trained neural network.

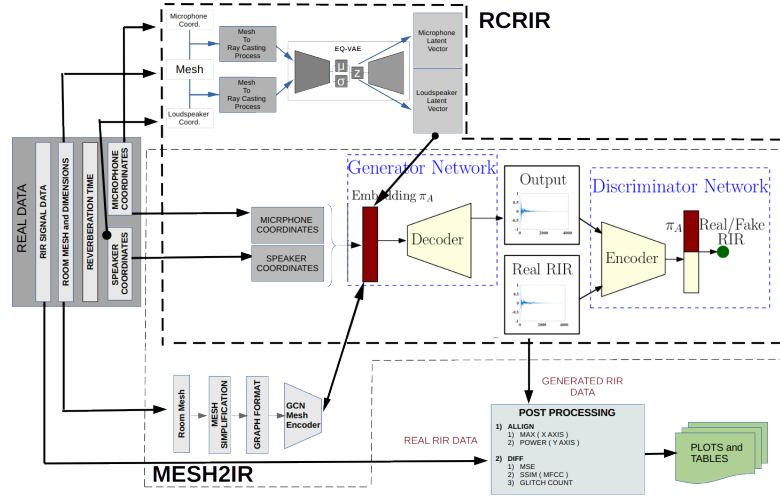


Figure 5.10: RCRIR model: The architecture enclosed by the dark dashed line corresponds to the proposed RCRIR model, whereas the architecture outlined with the light dashed line depicts the original MESH2IR model.

5.1.3. The DIMRIR Model

For comparative analysis and to evaluate the effectiveness of the proposed geometric models, we trained the acoustic model (StackGAN) using only the room dimensions as geometric input, without incorporating detailed spatial features. This baseline model, referred to as DIMRIR, uses a simplified representation of geometry based solely on room size parameters, as illustrated in Figure 5.11.

The generated latent vectors, combined with the microphone and loudspeaker coordinates, are input to the StackGAN acoustic model for RIR generation, as illustrated in Figure 5.10. Similar to the MESH-TRANSFORMER-AE (MESHTAE) model, the RCRIR architecture enables independent operation of the geometric and acoustic components, thereby eliminating the need to retrain the geometric model when encountering new room geometries.

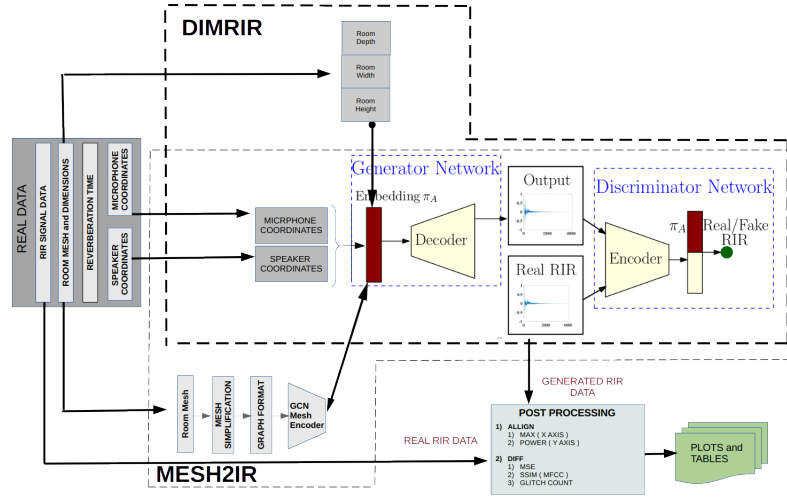


Figure 5.11: DIMRIR Model: The architecture enclosed by the dark dashed line corresponds to the DIMRIR model, whereas the architecture enclosed by the light dashed line corresponds to the MESH2IR model.

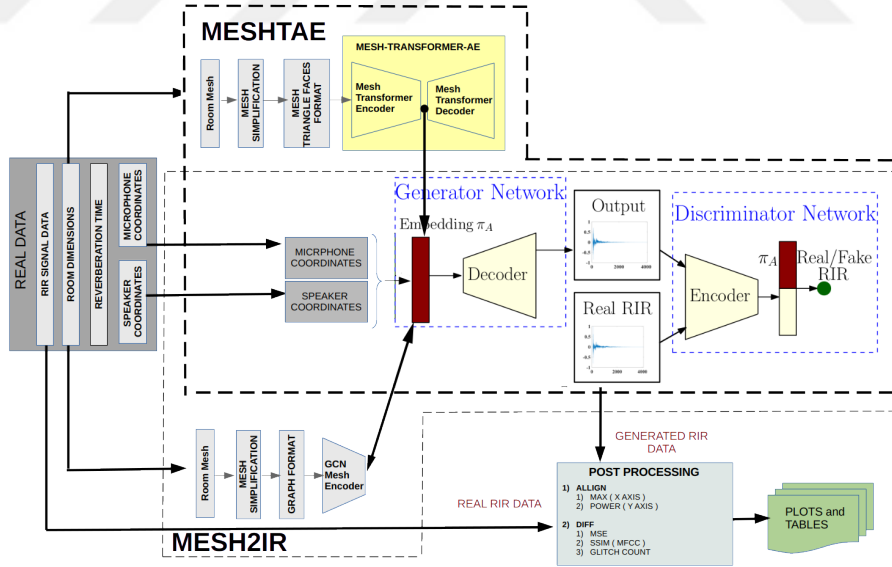


Figure 5.12: Comparison of RIRs generated by MESHTAE and MESH2IR [24] against real RIRs. The architecture enclosed by the dark dashed line corresponds to MESHTAE, while that enclosed by the light dashed line corresponds to MESH2IR.

Table 5.1: Model comparison table.

Test Dataset	Model	MEAN MSE	MEAN SSIM	MEAN GLITCH COUNT
GTU-RIR	DIMRIR	0.0737	0.1239	236.76
GTU-RIR	MESHTAE-500	0.0658	0.2348	167.94
GTU-RIR	MESHTAE-1500	0.0448	0.2661	131.74
GTU-RIR	RCRIR	0.0464	0.2861	127.36
GTU-RIR	MESH2IR	0.0436	0.2995	130.69
GTU-RIR	MESHTAE-2000	0.0363	0.3390	123.32
BUT ReverbDB	DIMRIR	0.0525	0.1954	563.67
BUT ReverbDB	RCRIR	0.0329	0.4356	259.26
BUT ReverbDB	MESH2IR	0.0283	0.4946	241.24
BUT ReverbDB	MESHTAE-500	0.0254	0.5296	168.15
BUT ReverbDB	MESHTAE-1500	0.0208	0.5760	134.82
BUT ReverbDB	MESHTAE-2000	0.0161	0.6401	103.08

5.2. Experiments and Evaluation

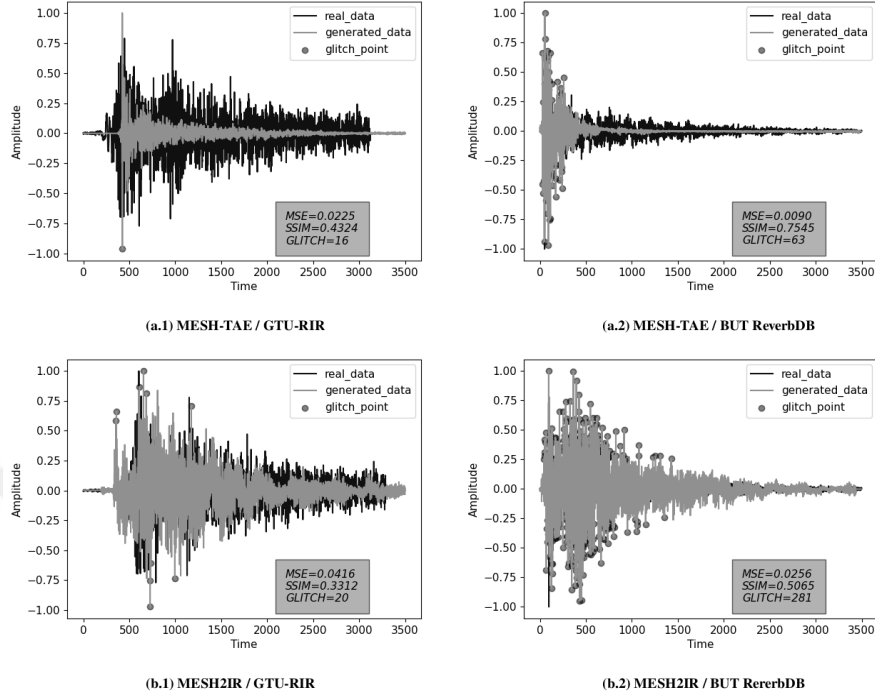


Figure 5.12 illustrates the process of comparing real and generated RIR signals for the MESHTAE model. In this comparison, both the MESH2IR and MESHTAE models generate the RIR signals by utilizing the loudspeaker and microphone coordinates along with the room mesh geometry as inputs. The corresponding processes for the DIMRIR and RCRIR models are depicted in Figure 5.11 and Figure 5.10, respectively.

The MESH2IR model, illustrated in Figure 5.12, incorporates a three-layer Graph Convolutional Network (GCN) termed MeshNet. To manage computational complexity, the input room mesh is down-sampled to 2,000 nodes prior to processing by the network. MeshNet comprises 3,560 trainable parameters, whereas the remaining components of the architecture—including the generator and discriminator modules from StackGAN-I—account for approximately 130 million parameters. The MESH2IR model receives microphone coordinates, loudspeaker coordinates, and the down-sampled room mesh as inputs, and produces the RIR signal as output, represented by a vector of 4,096 floating-point values.

The MESHTAE model, depicted in Figure 5.12, integrates a pre-trained Mesh Transformer Autoencoder (MESH-TRANSFORMER-AE). In a manner analogous to the MESH2IR model, the input room mesh is down-sampled to 2000 nodes to mitigate computational complexity. The Mesh Transformer Autoencoder, comprising approxi-

mately 537,000 parameters, is pretrained independently before its incorporation into the MESHTAE framework. Both the inputs and outputs of the MESHTAE model correspond closely with those of the MESH2IR model. For clarity, the two architectures are presented side-by-side in the figure, highlighting their shared structural features, including the adoption of the StackGAN-I model for the acoustic component.

The RCRIR model, depicted in Figure 5.10, utilizes a linear algorithm to generate two-dimensional ray casting images. These images are then encoded using EQ-VAE, a pre-trained variational autoencoder based on an enhanced Stable Diffusion VAE architecture, to extract latent vector representations. The original ray casting images have a resolution of $256 \times 256 \times 3$ pixels, which are compressed into latent vectors of size $16 \times 16 \times 3$. This latent vector serves as the geometric representation of the room and is subsequently used as input to the StackGAN acoustic model for RIR generation.

A total of twelve experiments were conducted utilizing the MESHTAE model (including its three variations), MESH2IR, RCRIR, and DIMRIR models. Training was performed on synthetic datasets, while evaluation was carried out using the real-world GTU-RIR and BUT ReverbDB datasets. The MESH2IR model was trained following the procedure outlined in its original study [24]. For the MESHTAE model, we first generated a synthetic dataset of room meshes, trained a mesh transformer autoencoder on this dataset, and subsequently employed the pretrained mesh transformer autoencoder to produce geometric embeddings during the training of the StackGAN acoustic model. The key differentiating factor among these models is that the MESHTAE approach leverages a fixed, pretrained geometric embedding model, which remains unchanged during the acoustic model training, unlike other models that require joint or repeated training of both components.

Evaluation of the results in the Table 5.1 :

1. MESHTAE: This model achieves the best performance in RIR generation when the number of mesh faces is set to 2000, as it retains more geometric detail compared to other configurations. Across all datasets, increasing the number of faces consistently improves RIR generation quality, indicating that more accurate geometric representations lead to better acoustic modeling. Furthermore, MESHTAE offers a significant advantage over MESH2IR: its geometric component—the mesh transformer autoencoder—can be pretrained independently, allowing for modular training and eliminating the need to retrain the geometric model alongside the acoustic model.
2. DIMRIR: This is the worst RIR generator as expected because we feed only room dimensions as the geometric information to the acoustic model that causes poor RIR generation.

3. DIMRIR: As expected, this model yields the poorest RIR generation performance among all approaches. This is primarily due to the limited geometric input—only the room dimensions—which provides insufficient spatial detail for accurate acoustic modeling.
4. MESH2IR: Serving as our baseline model, MESH2IR achieves the second-best performance among all approaches. While it effectively incorporates geometric information through a graph-based encoder, it requires retraining for each new room geometry, limiting its flexibility compared to MESHTAE.

5.3. Summary

This chapter explores the feasibility of decoupling the training of geometric and acoustic models in RIR generator GANs. Using synthetically generated geometric data, we trained a mesh transformer autoencoder to serve as the geometric model, enabling independence from specific room geometries. Experimental results, summarized in Table 5.1, demonstrate that separating the geometric and acoustic components is not only feasible but also effective. This decoupled approach allows the geometric model to be trained independently, offering greater modularity and flexibility when adapting to new room geometries, without requiring full model retraining.

The results confirm that using a pre-trained geometric model for embedding during acoustic model training does not degrade performance. Furthermore, the RCRIR model, which employs a linear ray casting process and a pre-trained EQ-VAE model for geometric representation, successfully generates RIRs that are comparable in quality to those produced by MESH2IR. Additionally, we evaluated a simplified model named DIMRIR, which uses only room dimensions as geometric input. As expected, DIMRIR yielded the lowest performance, reinforcing the importance of richer geometric representations for accurate RIR generation.

6. CONCLUSIONS

This research focused on RIR generation using generative adversarial neural networks (GANs), where the room geometry is represented as a mesh, and the microphone and loudspeaker locations are provided as inputs. We initially generated a real RIR dataset, called GTU-RIR, to test the neural network-based RIR generation against a dataset that is spatially more distributed. During the experiments, we observed that neural network-based room impulse response (RIR) generators sometimes introduce glitches into the generated RIRs, which cause distortions in the reverberant sound when convolved with audio signals. This issue arises because training neural networks with the Mean Squared Error (MSE) loss function minimizes the difference between the generated and real RIR data, but it does not account for the signal’s perceptual shape.

To mitigate these glitches, we adopted the approach of training GAN models with the Structural Similarity (SSIM) loss function in the generator networks. Training the neural network model with the SSIM loss function effectively reduced the number of glitches while preserving the MSE and SSIM values, demonstrating the benefits of using perceptual loss functions in RIR generation.

Furthermore, we identified a common limitation in neural network-based RIR generators, which often tightly integrate geometry embedding with RIR generation, reducing their flexibility. To address this, we decoupled the training of the geometric model and the acoustic model, allowing for greater flexibility in generating RIR signals for unseen geometries. We generated synthetic geometric data and trained the MESH-TRANSFORMER-AE geometric model on it to ensure geometric data independence. The experimental results, as shown in Table 5.1, indicate that it is indeed possible to decouple the geometric and acoustic components of RIR generator models.

This decoupling approach provides a significant advantage, as it enables the generation of latent vector representations of a reverberant environment without requiring training with actual sound data. By utilizing these latent vectors directly, we anticipate improvements in both the training process and the RIR generation process, reducing training time and increasing efficiency. This work sets the stage for more flexible and scalable RIR generation models that can handle a wide range of room geometries.

We also developed two additional models: RCRIR and DIMRIR. The RCRIR (Ray Casting RIR) model was designed to explore whether more realistic RIRs can be generated by providing possible sound propagation areas as 2D ray-casting images. Similar to the MESHTAE model, RCRIR separates the geometric and acoustic modeling components. While its results are close to those of the MESH2IR model, it does not outperform it. In contrast, the DIMRIR model was developed to assess the performance of RIR

generation when only room dimensions are provided as geometric input to the acoustic module. As expected, DIMRIR performs the worst among the MESH2IR, MESHTAE, and RCRIR models, as it supplies minimal geometric information to the acoustic model. Finally, the contributions and their possible application areas of this study can be summarized as follows:

1. **GTU-RIR Dataset and Open-Source Collection Framework:** We developed and publicly released a novel Room Impulse Response (RIR) dataset, named GTU-RIR, accompanied by complete technical documentation. This includes detailed schematics, bill of materials, hardware and software configurations, as well as the algorithms for semi-autonomous RIR collection and data cleaning. Unlike previous RIR datasets, our work is the first to openly share the full design and implementation details of the data collection system. Notably, the entire system was built using low-cost, off-the-shelf components, making it accessible for replication and further research. The open hardware and software framework can be readily adapted to collect underwater RIR data, supporting marine acoustics research and sonar-based applications.
2. **Minimization of Perceptual Artifacts Using SSIM in GAN Training:** We demonstrated that employing the Structural Similarity Index (SSIM) as a loss function in Generative Adversarial Networks (GANs) significantly reduces audible artifacts—particularly cracklings—in synthetically reverberated audio. This addresses a previously unexamined issue in RIR generation, where synthetic RIRs, despite being structurally similar, often introduced unnatural auditory artifacts. Our method yields more perceptually natural reverberation by reducing glitch content in the generated RIRs. To our knowledge, this is the first study to specifically target and minimize crackling artifacts in reverberated audio via loss function design in GAN-based RIR generation. GAN models trained with SSIM loss can be used to generate more realistic and immersive audio environments in virtual reality (VR) systems, reducing perceptual discomfort caused by unnatural reverberation.
3. **Geometry-Aware RIR Generation via Mesh Transformer Autoencoder:** We proposed a novel architecture that decouples geometric representation and RIR generation by employing a mesh transformer autoencoder. This modular approach enables independent training of geometry embeddings and acoustic modeling using synthetic data. Unlike traditional approaches that require joint training for each target geometry, our method allows pretraining of the geometric encoder and its direct integration into the RIR generation pipeline. This design facilitates generalization to previously unseen room configurations and significantly improves scalability. Furthermore, we validated the approach on both the GTU-RIR

dataset and the BUT ReverbDB, demonstrating its effectiveness across real and synthetic domains. While this work focuses on geometry-to-sound generation, the modularity of our framework enables future exploration in the reverse direction sound-to-geometry modeling. This could be extended to infer 3D room or underwater geometries from audio signals, with potential applications in acoustic scene reconstruction and sonar-based 3D mapping.

4. **Sound Propagation Field-Aware RIR Generation via Ray Casting:** We propose a novel architecture that incorporates sound propagation field information into the acoustic model using ray casting. This design performs comparably to previous models such as MESH2IR. However, RIR generation performance could potentially be improved by providing more detailed sound propagation data—such as third-order ray casting or 3D ray-casting information.

6.1. Future Work

It is recommended that future research continues to explore the interpretability and explainability of deep learning models in the context of RIR generation. Understanding how these models make decisions and what features influence their outputs could lead to improvements in model transparency and trustworthiness, which is crucial for their adoption in real-world applications. Additionally, further investigations can focus on expanding the application areas and datasets, considering emerging technologies and advancements in the field.

For future work, augmenting the room mesh with additional physical properties, such as sound reflection rates, is expected to enhance the fidelity and accuracy of the generated RIR signals. This would lead to more realistic and immersive auditory experiences, particularly in applications like virtual reality, acoustic simulations, and immersive sound systems. Including these additional physical properties could help generate more accurate reverberation characteristics and better model how sound interacts with different surfaces in a room.

Moreover, improving the geometric embedding model would further refine the RIR generation process. A promising avenue for improvement is to explore the integration of graph neural networks (GNNs) within the mesh transformer layers. GNNs have shown great potential in learning spatial relationships in graph-based data structures, and incorporating them into the mesh transformer could result in more robust geometric embeddings, improving the model’s ability to capture complex room geometries and acoustics. Such advancements will help ensure that RIR generation models can adapt more effectively to a wider variety of real-world environments, increasing their applicability across different industries.

Another possible improvement could involve providing more detailed sound propagation information to the acoustic model. For example, incorporating third-order reflection data as ray-casting field images might enhance performance. Alternatively, using 3D representations instead of 2D could also lead to better RIR generation results.



REFERENCES

- [1] Falcon Perez R., (2018), “Machine-learning-based estimation of room acoustic parameters”.
- [2] Türköz D. D., (2016), Comparison of single channel blind dereverberation methods for speech signals, PhD thesis.
- [3] Marquez-Borbon A., (2025), “Room impulse response measurement and analysis”, <https://ccrma.stanford.edu/~adnanm/SCI220/Music318ir.pdf> [Accessed: (February 2025)].
- [4] Foundation W., (2024), “Reverberation”, <https://en.wikipedia.org/wiki/Reverberation> [Accessed: (October 2024)].
- [5] Krasil’nikov V., Krylov V. V., (1984), “Introduction to physical acoustics”.
- [6] Christensen C. L., Koutsouris G., Rindel J. H., (2013), The iso 3382 parameters: Can we simulate them? can we measure them, in “Proceedings of the International Symposium on Room Acoustics, Toronto, ON, Canada”, Vol. 910.
- [7] Godin K., Gamper H., Raghuvanshi N., (2019), Aesthetic modification of room impulse responses for interactive auralization, in “Audio Engineering Society Conference: 2019 AES International Conference on Immersive and Interactive Audio”.
- [8] Wang X., Ghidaoui M., Lee P. J., (2018), Regularization for pipeline impulse response extraction with least square deconvolution, in “13th International Conference on Pressure Surges”, Bordeaux, France.
- [9] Gunel Kilic B., (2002), Room shape and size estimation using directional impulse response measurements, Forum Acusticum (16 - 20 Eylül 2002).
- [10] Yu W., Kleijn W. B., (2019), “Room geometry estimation from room impulse responses using convolutional neural networks”.
- [11] Song Q., Gu C., Tan R., (2018), “Deep room recognition using inaudible echos”, Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.
- [12] Li Q., Zhang X., Li H., (2018), Online direction of arrival estimation based on deep learning, in “2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, 2616–2620.
- [13] Papayiannis C., Evers C., Naylor P. A., (2019), “Detecting sound-absorbing materials in a room from a single impulse response using a crnn”.

- [14] Tervo S., Pätynen J., Lokki T., (2012), “Acoustic reflection localization from room impulse responses”, *Acta Acustica united with Acustica*, 98(3), 418–440.
- [15] Mitsuhashi K., Wang K., Anastasio M. A., (2014), “Investigation of the far-field approximation for modeling a transducer’s spatial impulse response in photoacoustic computed tomography”, *Photoacoustics*, 2(1), 21–32.
- [16] Allen J. B., Berkley D. A., (1979), “Image method for efficiently simulating small-room acoustics”, *The Journal of the Acoustical Society of America*, 65(4), 943–950.
- [17] Murphy D., Kelloniemi A., Mullen J., Shelley S., (2007), “Acoustic modeling using the digital waveguide mesh”, *IEEE Signal Processing Magazine*, 24(2), 55–66.
- [18] Hacıhabiboglu H., Gunel B., Cvetkovic Z., (2010), “Simulation of directional microphones in digital waveguide mesh-based models of room acoustics”, *IEEE Transactions on Audio, Speech, and Language Processing*, 18(2), 213–223.
- [19] Saarelma J., Greco G., (2015), Sound field visualization using the finite-difference time-domain method and measured spatial room impulse responses, in “Ninth International Conference On Auditorium Acoustics, Paris, France, October 29-31”, Institute of Acoustics. VK: Savioja, L.
- [20] Erkelens J. S., Heusdens R., (2011), A statistical room impulse response model with frequency dependent reverberation time for single-microphone late reverberation suppression, in “Twelfth Annual Conference of the International Speech Communication Association”.
- [21] Florencio D., Zhang Z., (2015), Maximum a posteriori estimation of room impulse responses, in “2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, 728–732.
- [22] Masztalski P., Matuszewski M., Piaskowski K., Romaniuk M., (2020), “Storir: Stochastic room impulse response generation for audio data augmentation”.
- [23] Ratnarajah A., Zhang S.-X., Yu M., Tang Z., Manocha D., Yu D., (2022a), Fast-rir: Fast neural diffuse room impulse response generator, in “ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, 571–575.
- [24] Ratnarajah A., Tang Z., Aralikatti R., Manocha D., (2022b), Mesh2ir: Neural acoustic impulse response generator for complex 3d scenes, in “Proceedings of the 30th ACM International Conference on Multimedia”, MM ’22, Association for Computing Machinery, New York, NY, USA, 924–933.
- [25] Ritter M., Mueller M., Stueker S., Metze F., Waibel A., (2016), Training deep neural

- networks for reverberation robust speech recognition, in “Speech Communication; 12. ITG Symposium”, 1–5.
- [26] Li C., Wang T., Xu S., Xu B., (2018), “Single-channel speech dereverberation via generative adversarial training”.
- [27] Larsson E., Viberg F., (2019), “Acoustic impulse response function estimation using machine learning”.
- [28] Singh N., Mentch J., Ng J., Beveridge M., Drori I., (2021), Image2reverb: Cross-modal reverb impulse response synthesis, in “Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)”, 286–295.
- [29] Ratnarajah A., Tang Z., Manocha D., (2021*a*), “Ir-gan: Room impulse response generator for far-field speech recognition”.
- [30] Ratnarajah A., Tang Z., Manocha D., (2021*b*), Ts-rir: Translated synthetic room impulse responses for speech augmentation, in “2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)”, 259–266.
- [31] Ratnarajah A., Manocha D., (2023), “Listen2Scene: Interactive material-aware binaural sound propagation for reconstructed 3D scenes”, arXiv e-prints, arXiv:2302.02809.
- [32] Ratnarajah A., Ghosh S., Kumar S., Chiniya P., Manocha D., (2024), Av-rir: Audio-visual room impulse response estimation, in “Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition”, 27164–27175.
- [33] Mehmet Pekmezci Y. G., (2024), “Gtu-rir git repository”, <https://github.com/mehmetpekmezci/gtu-rir/> [Accessed: (November 2024)].
- [34] Sabine W. C., (1927), “Collected papers on acoustics”, (No Title).
- [35] Rindel J. H., (2024), “Room acoustic measurement methods in the past, present and future, including the importance of the iso 3382 series”, *Journal Akustika*, 47(47), 3–13.
- [36] Escobar V. G., Morillas J. B., (2015), “Analysis of intelligibility and reverberation time recommendations in educational rooms”, *Applied Acoustics*, 96, 1–10.
- [37] Chen Z., (2007), “An investigation of acoustic impulse response measurement and modeling for small rooms”, Montana State University.
- [38] Fazenda B., Holland K., Newell P., Castro S., (1939), “Modulation transfer function as a measure of room low frequency”, *Journal of the Acoustical Society of America*, 11, 74.
- [39] Fausti P., Farina A., (2000), “Acoustic measurements in opera houses: compar-

- ison between different techniques and equipment”, *Journal of Sound and Vibration*, 232(1), 213–229.
- [40] Heyser R. C., (1967), “Acoustical measurements by time delay spectrometry”, *Journal of the Audio Engineering Society*, 15(4), 370–382.
- [41] SDR L., (2024), “Liquid sdr”, <https://liquidsdr.org/doc/msequence/> [Accessed: (November 2024)].
- [42] Orlando S., Peeters B., Coppotelli G., (2008), “Improved frf estimators for mimo sine sweep data”, 23rd International Conference on Noise and Vibration Engineering 2008, ISMA 2008.
- [43] Toma I. O., Budescu M., Albu G., (2009), “Seismic behaviour of an experimental model made of thin-walled cold formed steel profiles - hardell structures”, *Bulletin of the Polytechnic Institute of Jassy, CONSTRUCTIONS. ARCHITECTURE Section, Tomme LV (LIX), Fascicle 1*, pages 67-78 (2009).
- [44] Canfield-Dafilou E. K., Callery E., Jette C., (2018), A portable impulse response measurement system, in “15th Sound and Music Computing Conference”, 172–176.
- [45] Habets E. A., (2006), “Room impulse response generator”, Technische Universiteit Eindhoven, Tech. Rep, 2(2.4), 1.
- [46] Smith J. O., (1985), A new approach to digital reverberation using closed waveguide networks, in “Proc. Int. Computer Music Conf, Vancouver, 1985”, 47–53.
- [47] Savioja L., Rinne T., Takala T., (1994), Simulation of room acoustics with a 3-d finite difference mesh, in “The 1994 International Computer Music Conference, Aarhus, September 12-17, 1994”, International Computer Music Association ICMA, 463–466.
- [48] Atalay T. B., Gül Z. S., De Sena E., Cvetković Z., Hacıhabiboğlu H., (2022), “Scattering delay network simulator of coupled volume acoustics”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30, 582–593.
- [49] Papayiannis C., Evers C., Naylor P. A., (2019), “Data augmentation of room classifiers using generative adversarial networks”, arXiv preprint arXiv:1901.03257.
- [50] Lluís F., Martínez-Nuevo P., Bo Møller M., Ewan Shepstone S., (2020), “Sound field reconstruction in rooms: Inpainting meets super-resolution”, *The Journal of the Acoustical Society of America*, 148(2), 649–659.
- [51] Wager S., Choi K., Durand S., (2020), “Dereverberation using joint estimation of dry speech signal and acoustic system”, arXiv preprint arXiv:2007.12581.
- [52] Donahue C., McAuley J., Puckette M., (2019), “Adversarial audio synthesis”.

- [53] Zhu J.-Y., Park T., Isola P., Efros A. A., (2017), Unpaired image-to-image translation using cycle-consistent adversarial networks, in “Proceedings of the IEEE International Conference on Computer Vision (ICCV)”.
- [54] Tang Z., Chen L., Wu B., Yu D., Manocha D., (2020), Improving reverberant speech training using diffuse acoustic simulation, in “ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, 6969–6973.
- [55] ISO (2009), Acoustics — measurement of room acoustic parameters — part 1: Performance spaces., Standard, International Organization for Standardization, Geneva, CH.
- [56] Tang Z., Aralikatti R., Ratnarajah A. J., Manocha D., (2022), Gwa: A large high-quality acoustic dataset for audio processing, in “ACM SIGGRAPH 2022 Conference Proceedings”, SIGGRAPH ’22, Association for Computing Machinery, New York, NY, USA.
- [57] Schlichtkrull M., Kipf T. N., Bloem P., van den Berg R., Titov I., Welling M., (2018), Modeling relational data with graph convolutional networks, in A. Gangemi R. Navigli M.-E. Vidal P. Hitzler R. Troncy L. Hollink A. Tordai M. Alam, eds, “The Semantic Web”, Springer International Publishing, Cham, 593–607.
- [58] Ratnarajah A., Ananthabhotla I., Ithapu V. K., Hoffmann P., Manocha D., Calamia P., (2023), Towards improved room impulse response estimation for speech recognition, in “ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)”, 1–5.
- [59] Wang Z., Bovik A. C., Sheikh H. R., (2017), Structural similarity based image quality assessment, in “Digital Video image quality and perceptual coding”, CRC Press, 225–242.
- [60] Yeganeh H., (2014), Cross Dynamic Range And Cross Resolution Objective Image Quality Assessment With Applications, PhD thesis.
- [61] Wang Z., Bovik A., Sheikh H., Simoncelli E., (2004), “Image quality assessment: from error visibility to structural similarity”, IEEE Transactions on Image Processing, 13(4), 600–612.
- [62] Wang Z., Bovik A., (2002), “A universal image quality index”, IEEE Signal Processing Letters, 9(3), 81–84.
- [63] Van Dongen S., Enright A. J., (2012), “Metric distances derived from cosine similarity and pearson and spearman correlations”, arXiv preprint arXiv:1208.3145.
- [64] Mana S. C., Sasipraba T., (2021), Research on cosine similarity and pearson corre-

- lation based recommendation models, in “Journal of Physics: Conference Series”, Vol. 1770, IOP Publishing, 012014.
- [65] Terven J., Cordova-Esparza D. M., Ramirez-Pedraza A., Chavez-Urbiola E. A., Romero-Gonzalez J. A., (2023), “Loss functions and metrics in deep learning”, arXiv preprint arXiv:2307.02694.
- [66] Brunet D., Vrscay E. R., Wang Z., (2011), “On the mathematical properties of the structural similarity index”, IEEE Transactions on Image Processing, 21(4), 1488–1499.
- [67] Pekmezci M., Genc Y., (2024), “Evaluation of ssim loss function in rir generator gans”, Digital Signal Processing, 154, 104685.
- [68] Paszke A., Gross S., Massa F., Lerer A., Bradbury J., Chanan G., Killeen T., Lin Z., Gimelshein N., Antiga L., Desmaison A., Kopf A., Yang E., DeVito Z., Raison M., Tejani A., Chilamkurthy S., Steiner B., Fang L., Bai J., Chintala S., (2019), Pytorch: An imperative style, high-performance deep learning library, in H. Wallach H. Larochelle A. Beygelzimer F. d'Alché-Buc E. Fox R. Garnett, eds, “Advances in Neural Information Processing Systems 32”, Curran Associates, Inc., 8024–8035.
- [69] Pambrun J.-F., Noumeir R., (2015), Limitations of the ssim quality metric in the context of diagnostic imaging, in “2015 IEEE international conference on image processing (ICIP)”, IEEE, 2960–2963.
- [70] Nilsson J., Akenine-Möller T., (2020), “Understanding ssim”, arXiv preprint arXiv:2006.13846.
- [71] Wang Z., Simoncelli E. P., Bovik A. C., (2003), Multiscale structural similarity for image quality assessment, in “The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003”, Vol. 2, Ieee, 1398–1402.
- [72] Sampat M. P., Wang Z., Gupta S., Bovik A. C., Markey M. K., (2009), “Complex wavelet structural similarity: A new image similarity index”, IEEE transactions on image processing, 18(11), 2385–2401.
- [73] Zeng K., Wang Z., (2012), 3d-ssim for video quality assessment, in “2012 19th IEEE international conference on image processing”, IEEE, 621–624.
- [74] Chen S., Zhang Y., Li Y., Chen Z., Wang Z., (2018), Spherical structural similarity index for objective omnidirectional video quality assessment, in “2018 IEEE international conference on multimedia and expo (ICME)”, IEEE, 1–6.
- [75] Ashesh A., Deschamps J., Jug F., (2024), “Microsim: Improved structural similarity for comparing microscopy data”, arXiv preprint arXiv:2408.08747.

- [76] Hassan M., Bhagvati C., (2012), “Structural similarity measure for color images”, *International Journal of Computer Applications*, 43(14), 7–12.
- [77] Kolaman A., Yadid-Pecht O., (2011), “Quaternion structural similarity: a new quality index for color images”, *IEEE Transactions on Image Processing*, 21(4), 1526–1536.
- [78] Venkataramanan A. K., Wu C., Bovik A. C., Katsavounidis I., Shahid Z., (2021), “A hitchhiker’s guide to structural similarity”, *IEEE Access*, 9, 28872–28896.
- [79] Jaafar M. I., Nawawi S. W., Rahim R. A., (2022), “Improving measurement bias of structural similarity index (ssim) using absolute difference equation”, *Applications of Modelling and Simulation*, 6, 10–19.
- [80] Sasaki T., Fukushima N., Maeda Y., Sugimoto K., Kamata S.-i., (2020), Constant-time gaussian filtering for acceleration of structure similarity, in “2020 International Conference on Image Processing and Robotics (ICIP)”, IEEE, 1–6.
- [81] Bakurov I., Buzzelli M., Schettini R., Castelli M., Vanneschi L., (2022), “Structural similarity index (ssim) revisited: A data-driven approach”, *Expert Systems with Applications*, 189, 116087.
- [82] Breakey D., Meskell C., (2013), “Comparison of metrics for the evaluation of similarity in acoustic pressure signals”, *Journal of Sound and Vibration*, 332(15), 3605–3609.
- [83] Kandadai S., Hardin J., Creusere C. D., (2008), Audio quality assessment using the mean structural similarity measure, in “2008 IEEE International Conference on Acoustics, Speech and Signal Processing”, 221–224.
- [84] Gan C., Wang X., Zhu M., Yu X., (2011), Audio quality evaluation using frequency structural similarity measure, in “IET International Communication Conference on Wireless Mobile and Computing (CCWMC 2011)”, 299–303.
- [85] Zhu M., Zheng J., Jin C., Wan W., (2013), Structural similarity analysis of modulation for audio quality assessment, in “2013 IEEE International Conference on Acoustics, Speech and Signal Processing”, IEEE, 403–407.
- [86] Sheng L., Huang D.-Y., Pavlovskiy E. N., (2019), “High-quality speech synthesis using super-resolution mel-spectrogram”, *arXiv preprint arXiv:1912.01167*.
- [87] Breakey D., Meskell C., (2013), “Comparison of metrics for the evaluation of similarity in acoustic pressure signals”, *Journal of Sound and Vibration*, 332(15), 3605–3609.
- [88] Zhao H., Gallo O., Frosio I., Kautz J., (2017), “Loss functions for image restoration with neural networks”, *IEEE Transactions on Computational Imaging*, 3(1), 47–57.
- [89] Nilsson J., Akenine-Möller T., (2020), “Understanding ssim”.

- [90] Mildenhall B., Srinivasan P. P., Tancik M., Barron J. T., Ramamoorthi R., Ng R., (2021), “Nerf: Representing scenes as neural radiance fields for view synthesis”, *Communications of the ACM*, 65(1), 99–106.
- [91] Kipf T. N., Welling M., (2016a), “Variational graph auto-encoders”, arXiv preprint arXiv:1611.07308.
- [92] Kipf T. N., Welling M., (2016b), “Semi-supervised classification with graph convolutional networks”, arXiv preprint arXiv:1609.02907.
- [93] Zhou J., Cui G., Hu S., Zhang Z., Yang C., Liu Z., Wang L., Li C., Sun M., (2020), “Graph neural networks: A review of methods and applications”, *AI open*, 1, 57–81.
- [94] Wu Z., Pan S., Chen F., Long G., Zhang C., Philip S. Y., (2020), “A comprehensive survey on graph neural networks”, *IEEE transactions on neural networks and learning systems*, 32(1), 4–24.
- [95] Litany O., Bronstein A., Bronstein M., Makadia A., (2018), Deformable shape completion with graph convolutional autoencoders, in “Proceedings of the IEEE conference on computer vision and pattern recognition”, 1886–1895.
- [96] Simonovsky M., Komodakis N., (2018), Graphvae: Towards generation of small graphs using variational autoencoders, in “Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part I 27”, Springer, 412–422.
- [97] Salha G., Hennequin R., Vazirgiannis M., (2019), “Keep it simple: Graph autoencoders without graph convolutional networks”, arXiv preprint arXiv:1910.00942.
- [98] Li J., Yu T., Juan D.-C., Gopalan A., Cheng H., Tomkins A., (2020), “Graph autoencoders with deconvolutional networks”, arXiv preprint arXiv:2012.11898.
- [99] Li J., Li J., Liu Y., Yu J., Li Y., Cheng H., (2021), “Deconvolutional networks on graph data”, *Advances in Neural Information Processing Systems*, 34, 21019–21030.
- [100] Salha G., Hennequin R., Vazirgiannis M., (2021), Simple and effective graph autoencoders with one-hop linear models, in “Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020, Ghent, Belgium, September 14–18, 2020, Proceedings, Part I”, Springer, 319–334.
- [101] Ge Y., Pang Y., Li L., Itti L., (2021), Graph autoencoder for graph compression and representation learning, in “Neural Compression: From Information Theory to Applications–Workshop@ ICLR 2021”.
- [102] Saint A., Shabayek A., Cherenkova K., Das R., Gusev G., Aouada D., Ottersten B., (2018), “Deep learning advances on different 3d data representations: A survey”.

- [103] Zhang C.-Y., Hu J., Yang L., Chen C. P., Yao Z., (2020), “Graph deconvolutional networks”, *Information Sciences*, 518, 330–340.
- [104] Yun S., Jeong M., Kim R., Kang J., Kim H. J., (2019), “Graph transformer networks”, *Advances in neural information processing systems*.
- [105] Mialon G., Chen D., Selosse M., Mairal J., (2021), “Graphit: Encoding graph structure in transformers”, *arXiv preprint arXiv:2106.05667*.
- [106] Lin K., Wang L., Liu Z., (2021), End-to-end human pose and mesh reconstruction with transformers, in “*Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*”, 1954–1963.
- [107] Zhang S., Chen H., Yang H., Sun X., Yu P., Xu G., (2022), “Graph masked autoencoders with transformers. arxiv 2022”, *arXiv preprint arXiv:2202.08391*.
- [108] Khajenezhad A., Osia S. A., Karimian M., Beigy H., (2022), “Gransformer: Transformer-based graph generation”, *arXiv preprint arXiv:2203.13655*.
- [109] Rampásek L., Galkin M., Dwivedi V. P., Luu A. T., Wolf G., Beaini D., (2022), “Recipe for a general, powerful, scalable graph transformer”, *Advances in Neural Information Processing Systems*, 35, 14501–14515.
- [110] Chen D., O’Bray L., Borgwardt K., (2022), Structure-aware transformer for graph representation learning, in “*International Conference on Machine Learning*”, PMLR, 3469–3489.
- [111] Nguyen D. Q., Nguyen T. D., Phung D., (2022), Universal graph transformer self-attention networks, in “*Companion Proceedings of the Web Conference 2022*”, 193–196.
- [112] Vecchio G., Prezzavento L., Pino C., Rundo F., Palazzo S., Spampinato C., (2023), “Met: A graph transformer for semantic segmentation of 3d meshes”, *Computer Vision and Image Understanding*, 235, 103773.
- [113] Müller L., Galkin M., Morris C., Rampásek L., (2023), “Attending to graph transformers”, *arXiv preprint arXiv:2302.04181*.
- [114] Gou M., Xiong F., Camps O., Sznai M., (2018), Monet: Moments embedding network, in “*Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*”, 3175–3183.
- [115] Feng Y., Feng Y., You H., Zhao X., Gao Y., (2019), Meshnet: Mesh neural network for 3d shape representation, in “*Proceedings of the AAAI conference on artificial intelligence*”, Vol. 33, 8279–8286.

- [116] Tang J., Zhou H., Chen X., Hu T., Ding E., Wang J., Zeng G., (2023), Delicate textured mesh recovery from nerf via adaptive surface refinement, in “Proceedings of the IEEE/CVF International Conference on Computer Vision”, 17739–17749.
- [117] Rakotosaona M.-J., Manhardt F., Arroyo D. M., Niemeyer M., Kundu A., Tombari F., (2024), Nerfmeshing: Distilling neural radiance fields into geometrically-accurate 3d meshes, in “2024 International Conference on 3D Vision (3DV)”, IEEE, 1156–1165.
- [118] Liang Y., Zhao S., Yu B., Zhang J., He F., (2022), Meshmae: Masked autoencoders for 3d mesh data analysis, in “European Conference on Computer Vision”, Springer, 37–54.
- [119] Siddiqui Y., Alliegro A., Artemov A., Tommasi T., Sirigatti D., Rosov V., Dai A., Nießner M., (2024), Meshgpt: Generating triangle meshes with decoder-only transformers, in “Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition”, 19615–19625.
- [120] Haag J., (2024), “Realrirs”, <https://github.com/jonashaag/RealRIRs/> [Accessed: (November 2024)].
- [121] James R., (2024), “Room impulse response datasets”, <https://github.com/RoyJames/room-impulse-responses/> [Accessed: (November 2024)].
- [122] Farina A., (2024), “Impulse response measurements, nordic sound symposium xxiii, bolkesjø, norway”.
- [123] Everest F. A., (2022), “Master handbook of acoustics”.
- [124] Alkin O., (2014), “Signals and systems: a MATLAB® integrated approach”, CRC press.
- [125] Oppenheim A. V., Willsky A. S., Nawab S. H., (1997), “Signals & systems”, Pearson Educación.
- [126] Jordan K. L., (1961), “Discrete representations of random signals”.
- [127] Foundation W., (2024), “Lti”, https://en.wikipedia.org/wiki/Linear_time-invariant_system [Accessed: (October 2024)].
- [128] Pelling A., Sarradj E., (2021), “Efficient forced response computations of acoustical systems with a state-space approach. acoustics 2021, 3, 581–593”.
- [129] Mourjopoulos J., (1985), “On the variation and invertibility of room impulse response functions”, Journal of Sound and Vibration, 102(2), 217–228.
- [130] McLachlan N. W., (1934), “Loud speakers: Theory, performance, testing and design”, (No Title).

- [131] Schaumberger A., (1971), “Impulse measurement techniques for quality determination in hi-fi equipment, with special emphasis on loudspeakers”, *Journal of the Audio Engineering Society*, 19(2), 101–107.
- [132] Carini A., Cecchi S., Orcioni S., (2020), “Robust room impulse response measurement using perfect periodic sequences for wiener nonlinear filters”, *Electronics*.
- [133] Chu W., (1990), “Impulse-response and reverberation-decay measurements made by using a periodic pseudorandom sequence”, *Applied Acoustics*, 29(3), 193–205.
- [134] Heyser R. C., (1973), “Time Delay Spectrometry”, *The Journal of the Acoustical Society of America*, 53(1), 319–320.
- [135] Farina A., (2000), “Simultaneous measurement of impulse response and distortion with a swept-sine technique”, *Journal of The Audio Engineering Society*.
- [136] Meng Q., Sen D., Wang S., Hayes L., (2008), Impulse response measurement with sine sweeps and amplitude modulation schemes, in “2008 2nd International Conference on Signal Processing and Communication Systems”, *IEEE*, 1–5.
- [137] Barré S., Ortiz N. M., (2015), Room impulse response measurement and delay-and-sum beamforming, application to room and building acoustics, in “Proc. Euronoise”, Vol. 2015, 161–166.
- [138] Foundation W., (2025), “Fourier transform”, https://en.wikipedia.org/wiki/Fourier_transform [Accessed: (February 2025)].
- [139] Stan G.-B., Embrechts J.-J., Archambeau D., (2002), “Comparison of different impulse response measurement techniques”, *Journal of the Audio engineering society*, 50(4), 249–262.
- [140] Nussbaumer H. J., Nussbaumer H. J., (1982), “The fast Fourier transform”, Springer.
- [141] Vetter K., Di Rosario S., (2011), Exochirptoolbox: a pure data implementation of ess impulse response measurement, in “Pure Data Convention”.
- [142] Mateljan I., Ugrinović K., (2003), The comparison of room impulse response measuring systems, in “Proceedings of the First Congress of Alps Adria Acoustics Association”, Citeseer.
- [143] Peng L., (2017), 13 - sound absorption and insulation functional composites, in M. Fan F. Fu, eds, “Advanced High Strength Natural Fibre Composites in Construction”, Woodhead Publishing, 333–373.
- [144] Amares S., Sujatmika E., Hong T. W., Durairaj R., Hamid H. S. H. B., (2017), “A

- review: Characteristics of noise absorption material”, *Journal of Physics: Conference Series*, 908(1), 012005.
- [145] Jambrosic K., Horvat M., Domitrovic H., (2008), “Reverberation time measuring methods”, *Journal of the Acoustical Society of America*, 123(5), 3617–3617.
- [146] Expert A., (2024), “Common absorption coefficients for acoustical treatments”, <https://commercial-acoustics.com/common-absorption-coefficients-for-acoustical-treatments/> [Accessed: (November 2024)].
- [147] Novak A., Simon L., Lotton P., (2015), “Synchronized Swept-Sine: Theory, Application, and Implementation”, *Journal of the Audio Engineering Society*, 63(10), 786–798.
- [148] Schroeder M. R., (2005), “New Method of Measuring Reverberation Time”, *The Journal of the Acoustical Society of America*, 37(3), 409–412.
- [149] Por E., van Kooten M., Sarkovic V., (2019), “Nyquist–shannon sampling theorem”, *Leiden University*, 1(1), 1–2.
- [150] Zhang H., Xu T., Li H., Zhang S., Wang X., Huang X., Metaxas D. N., (2017), Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks, in “Proceedings of the IEEE International Conference on Computer Vision (ICCV)”.
- [151] Kouzelis T., Kakogeorgiou I., Gidaris S., Komodakis N., (2025), “Eq-vae: Equivariance regularized latent space for improved generative image modeling”, *arXiv preprint arXiv:2502.09509*.
- [152] Szoke I., Skacel M., Mosner L., Paliesek J., Cernocky J. H., (2018), “Building and evaluation of a real room impulse response dataset”, *IEEE Journal of Selected Topics in Signal Processing*, 13, 863–876.
- [153] Pandkar M., (2025), “2d ray casting”, <https://github.com/MugPand/2D-Ray-Casting> [Accessed: (February 2025)].

BIOGRAPHY

Mehmet PEKMEZCİ received the BS degree in computer engineering from the Galatasaray University (2003) and the MS degree in computer science from the Maltepe University (2012). His research interests include audio signal processing, deep learning.



PUBLICATIONS WITHIN the SCOPE of THESIS STUDY

[1] Pekmezci M., Genc Y., (2024), “Evaluation of ssim loss function in rir generator gans”, Digital Signal Processing, 154, 104685.



APPENDICES

Appendice-A : Survey on Perceived Quality of Generated RIRs



RIR Human Evaluation Questionnaire

To evaluate the RIRs generated by 7 different neural network models, we selected random audio samples from the GTU-RIR dataset, which was recorded in 11 different rooms within the GTU Computer Engineering Department.

For each room, we provided the participants with the real song recorded using the selected microphone in that room. In addition to this, we provided the same song reverberated using both the real RIR and synthetic RIRs generated by the models.

We ask to rate the similarity between the real recorded song and the artificially reverberated versions (using both real and synthetic RIRs) on a scale from 1 to 10, where a score of 1 indicates the least similarity and a score of 10 indicates the highest similarity :

ROOM : room-207
Room Dimensions - Width, Depth, Height (m) : 11.27x5.42x2.83
Total Room Volume (m³) : 172.9
Total Room Area (m²) : 216.6
Speaker Coordinates X,Y,Z (m) : 3.01x4.25x1.67
Microphone Coordinates X,Y,Z (m) : 0.90x10.39x2.07

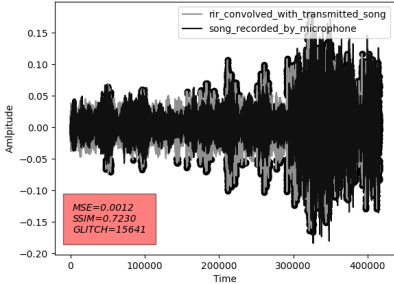
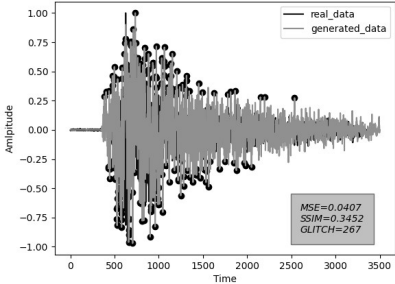
TYPE	RIR COHERENCE PLOT	SONG COHERENCE PLOT	AUDIO	RATING
Real Reverberated Song Recorded Using Microphone	-	-	0:00 / 0:26	10
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal			0:00 / 0:26	8
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model			0:00 / 0:26	6
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model			0:00 / 0:26	9

Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model				9
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model				7
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model				8
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model				9
Artificially Reverberated Song Using RIR Generated by MESHTAE Model				9

ROOM : room-208
Room Dimensions - Width, Depth, Height (m) : 5.32x9.47x2.9
Total Room Volume (m³) : 146.1
Total Room Area (m²) : 186.5
Speaker Coordinates X,Y,Z (m) : 6.43x3.11x1.89
Microphone Coordinates X,Y,Z (m) : 2.82x1.02x1.57

TYPE	RIR COHERENCE PLOT	SONG COHERENCE PLOT	AUDIO	RATING
Real Reverberated Song Recorded Using Microphone	-	-	▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal			▶ 0:00 / 0:26 — 🔊 ⋮	9
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	6
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model			▶ 0:00 / 0:26 — 🔊 ⋮	7
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	7
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	7

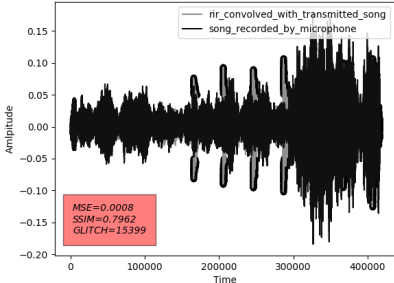
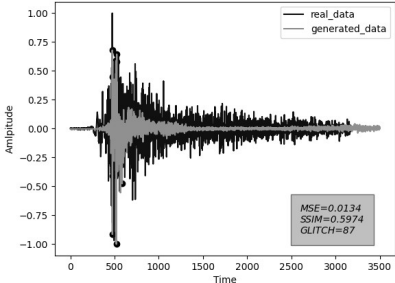
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:26

8

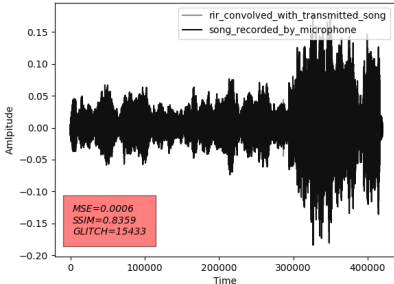
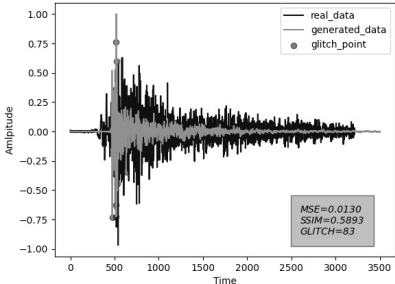
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:26

7

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



0:00 / 0:26

8

ROOM : room-conference01
Room Dimensions - Width, Depth, Height (m) : 11.85x6.0x2.95
Total Room Volume (m^3) : 209.7
Total Room Area (m^2) : 247.5
Speaker Coordinates X,Y,Z (m) : 3.43x2.37x2.10
Microphone Coordinates X,Y,Z (m) : 2.00x5.77x1.55

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

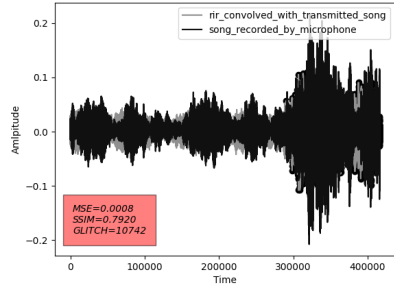
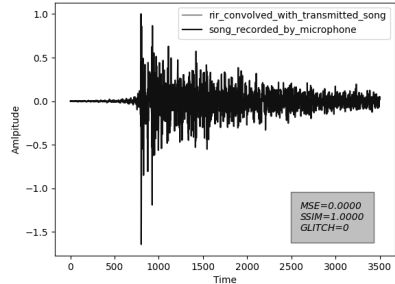
-

-

0:00 / 0:26

10

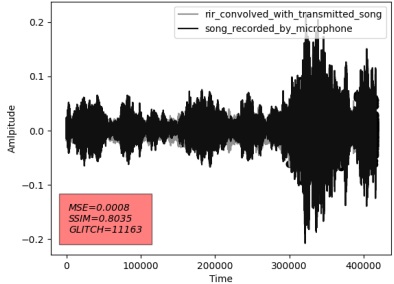
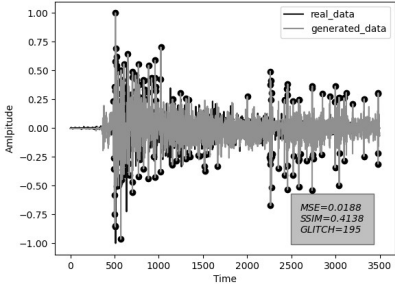
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



0:00 / 0:26

9

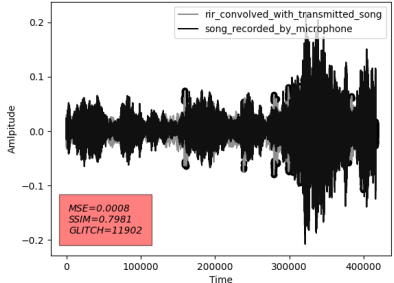
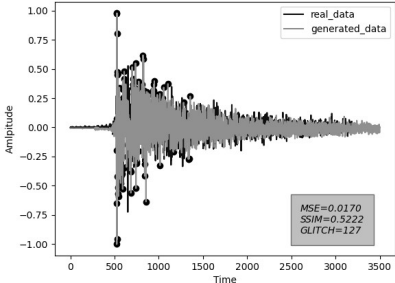
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model



0:00 / 0:26

6

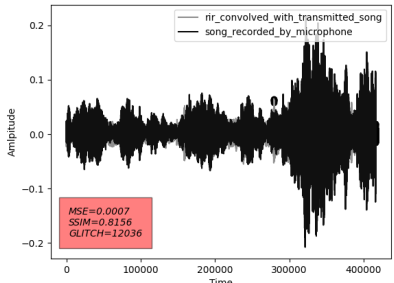
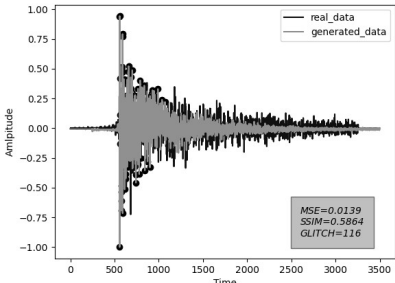
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



0:00 / 0:26

7

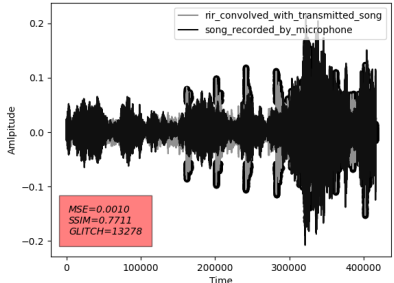
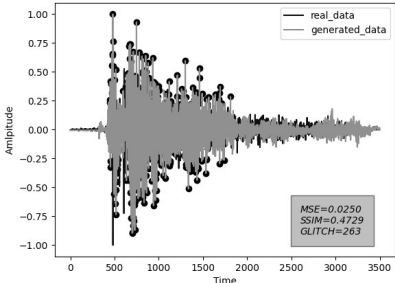
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model



0:00 / 0:26

7

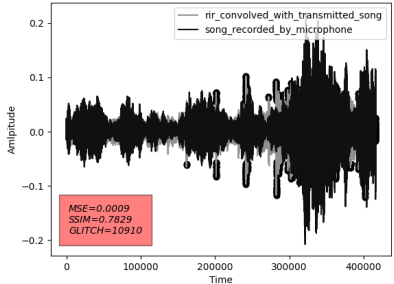
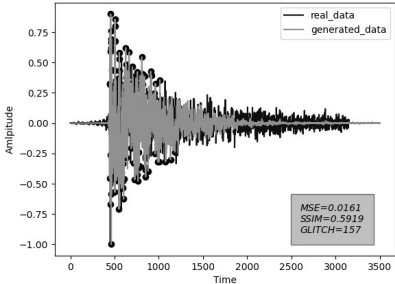
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model



0:00 / 0:26

6

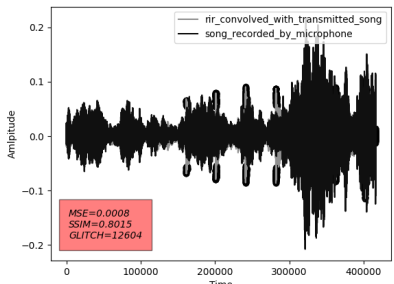
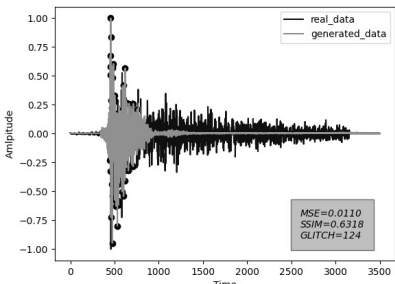
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:26

6

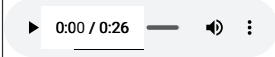
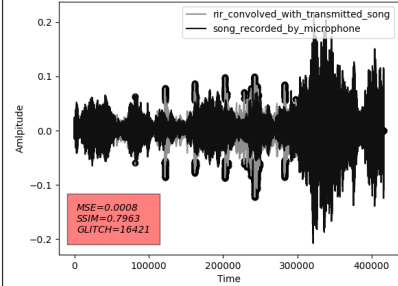
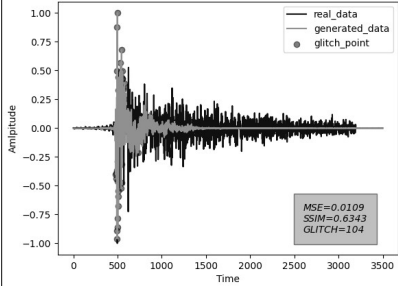
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:26

7

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



7

ROOM : room-sport01
Room Dimensions - Width, Depth, Height (m) : 7.68x11.95x3.96
Total Room Volume (m^3) : 363.4
Total Room Area (m^2) : 339.0
Speaker Coordinates X,Y,Z (m) : 2.06x4.76x2.05
Microphone Coordinates X,Y,Z (m) : 6.09x5.51x2.15

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

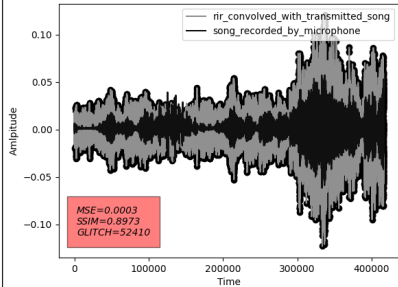
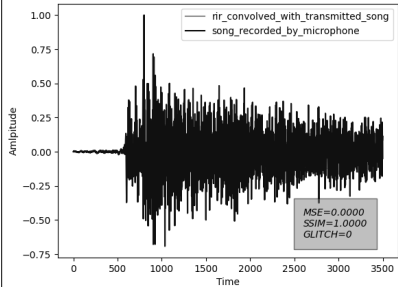
-

-



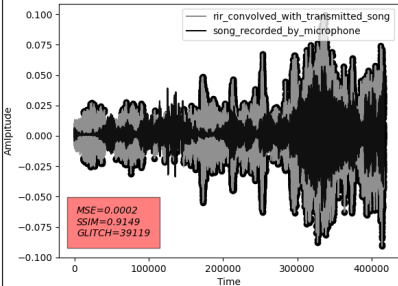
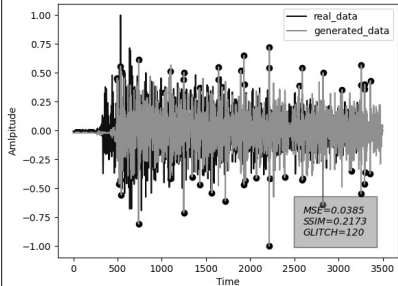
10

Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



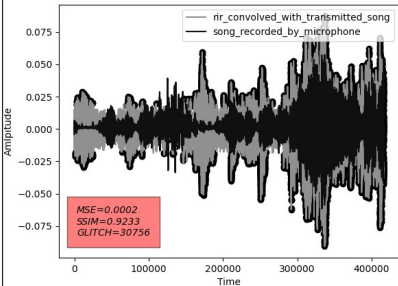
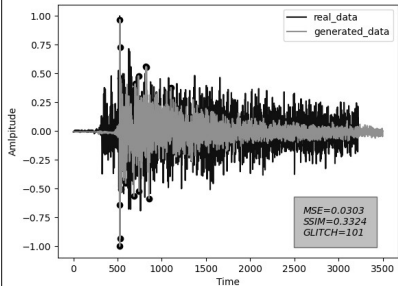
9

Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model

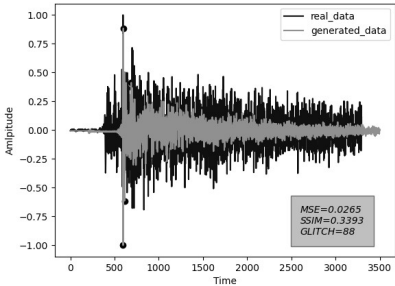
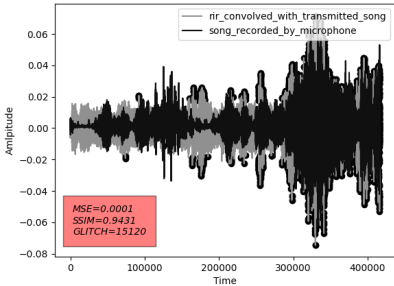
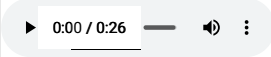
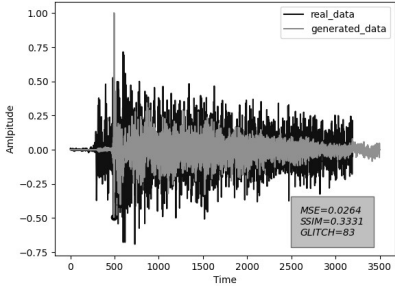
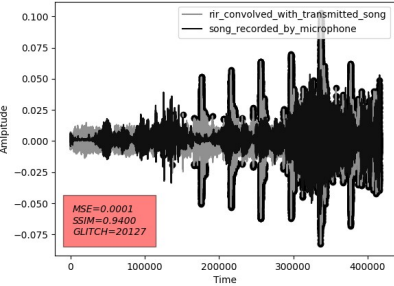
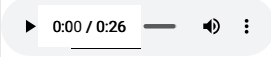
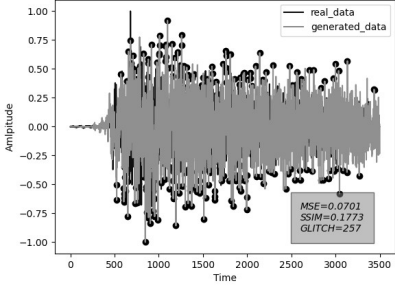
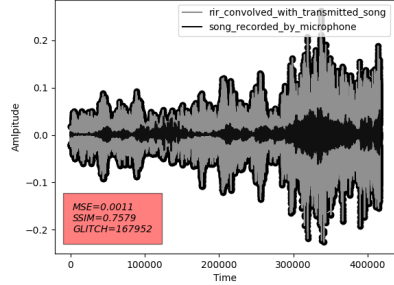
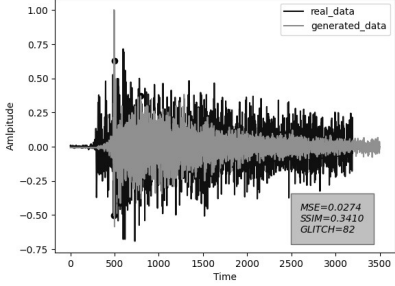
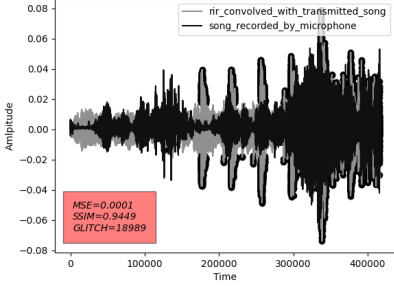
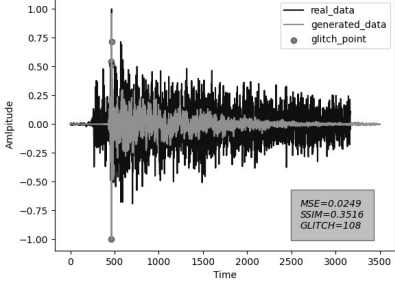
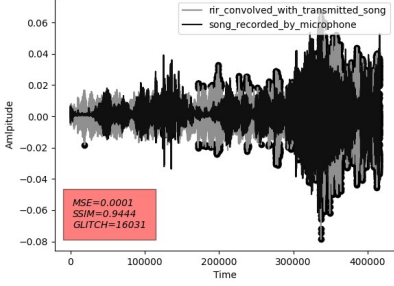
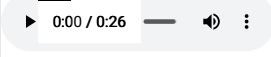


6

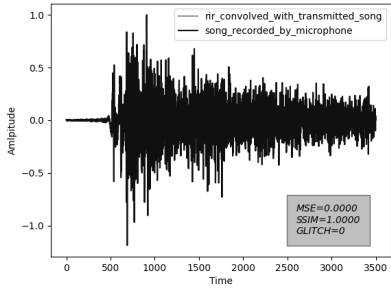
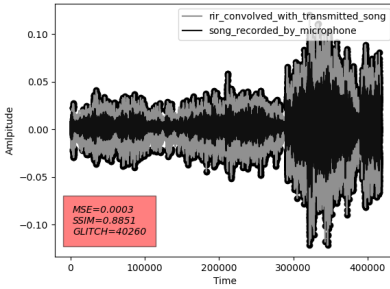
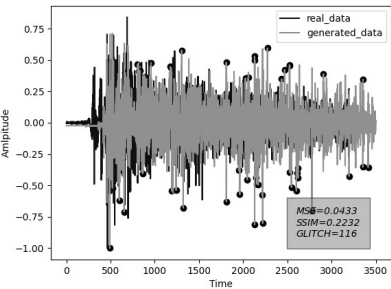
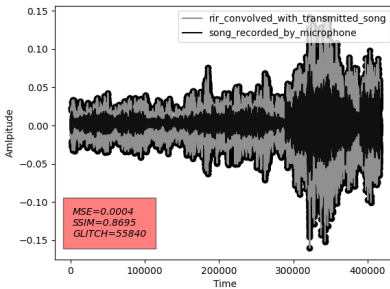
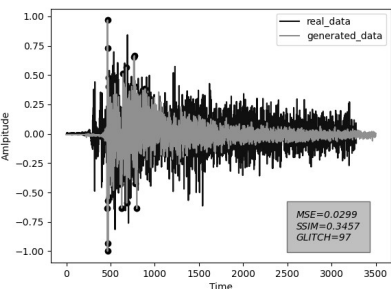
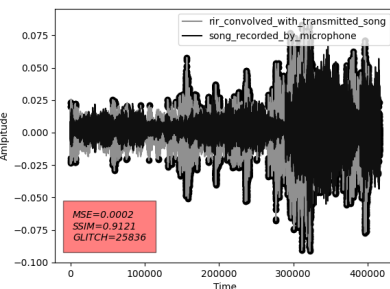
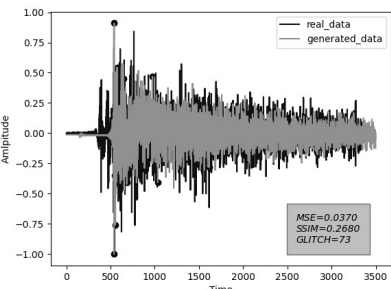
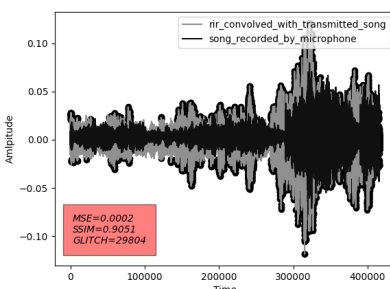
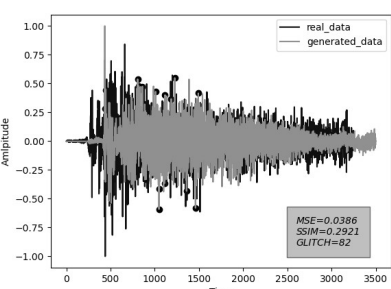
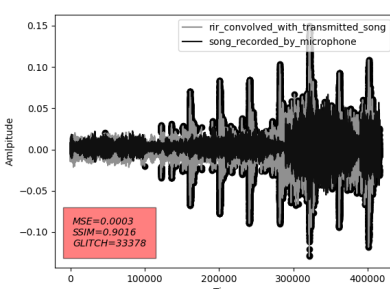
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



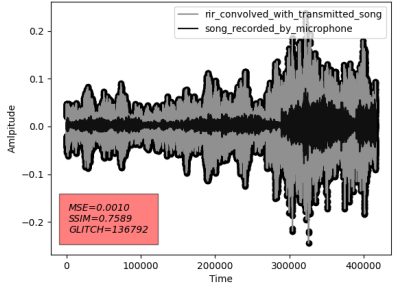
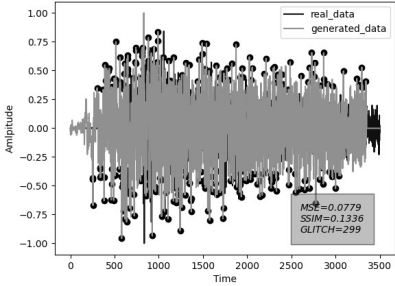
7

Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model				8
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model				6
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model				6
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model				7
Artificially Reverberated Song Using RIR Generated by MESHTAE Model				8

ROOM : room-sport02
Room Dimensions - Width, Depth, Height (m) : 6.79x7.97x2.96
Total Room Volume (m³) : 160.2
Total Room Area (m²) : 195.6
Speaker Coordinates X,Y,Z (m) : 1.93x4.16x2.05
Microphone Coordinates X,Y,Z (m) : 6.10x4.54x2.15

TYPE	RIR COHERENCE PLOT	SONG COHERENCE PLOT	AUDIO	RATING
Real Reverberated Song Recorded Using Microphone	-	-	▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal	 Amplitude vs Time (0-3500). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0000, SSIM=1.0000, GLITCH=0.	 Amplitude vs Time (0-400000). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0003, SSIM=0.8851, GLITCH=40260.	▶ 0:00 / 0:26 — 🔊 ⋮	9
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black line), generated_data (grey line). Metrics: MSE=0.0433, SSIM=0.2232, GLITCH=116.	 Amplitude vs Time (0-400000). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0004, SSIM=0.8695, GLITCH=55840.	▶ 0:00 / 0:26 — 🔊 ⋮	8
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model	 Amplitude vs Time (0-3500). Legend: real_data (black line), generated_data (grey line). Metrics: MSE=0.0299, SSIM=0.3457, GLITCH=97.	 Amplitude vs Time (0-400000). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0002, SSIM=0.9121, GLITCH=25836.	▶ 0:00 / 0:26 — 🔊 ⋮	7
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black line), generated_data (grey line). Metrics: MSE=0.0370, SSIM=0.2680, GLITCH=73.	 Amplitude vs Time (0-400000). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0002, SSIM=0.9051, GLITCH=29804.	▶ 0:00 / 0:26 — 🔊 ⋮	7
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black line), generated_data (grey line). Metrics: MSE=0.0386, SSIM=0.2921, GLITCH=82.	 Amplitude vs Time (0-400000). Legend: rir_convolved_with_transmitted_song (black line), song_recorded_by_microphone (grey line). Metrics: MSE=0.0003, SSIM=0.9016, GLITCH=33378.	▶ 0:00 / 0:26 — 🔊 ⋮	8

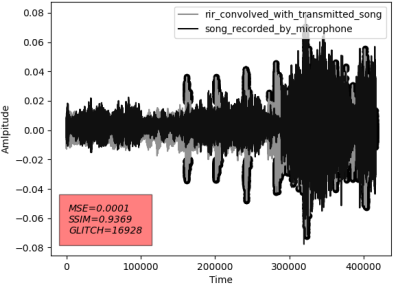
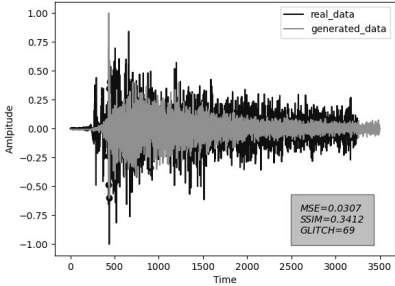
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:26

5

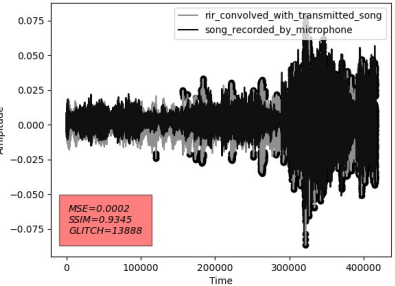
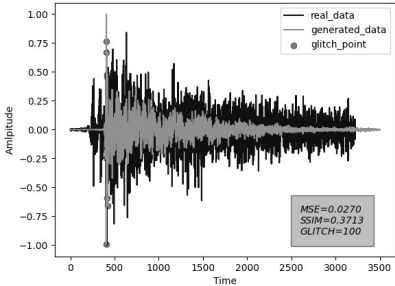
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:26

7

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



0:00 / 0:26

7

ROOM : room-z02
Room Dimensions - Width, Depth, Height (m) : 8.25x5.15x3.29
Total Room Volume (m³) : 139.8
Total Room Area (m²) : 173.1
Speaker Coordinates X,Y,Z (m) : 2.80x2.45x1.99
Microphone Coordinates X,Y,Z (m) : 3.56x5.25x2.13

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

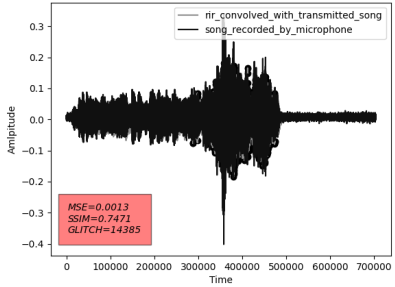
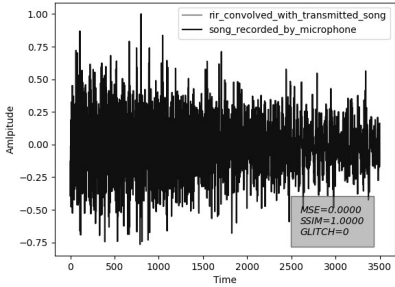
-

-

0:00 / 0:43

10

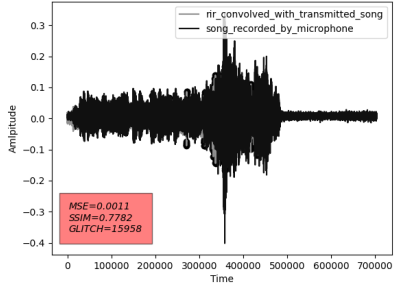
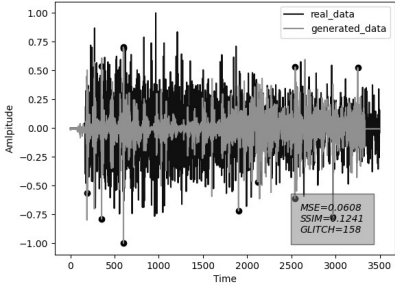
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



0:00 / 0:43

6

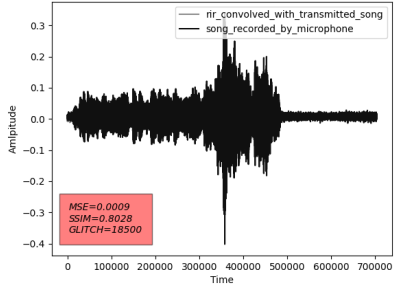
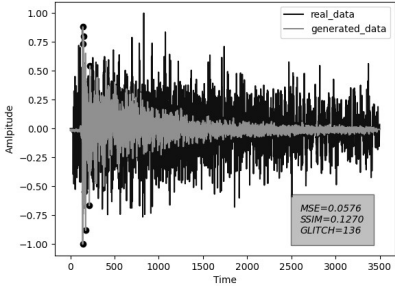
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model



0:00 / 0:43

7

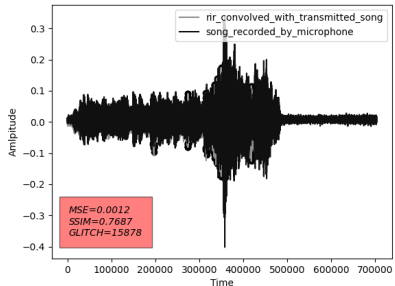
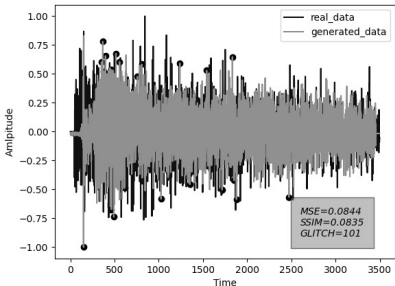
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



0:00 / 0:43

7

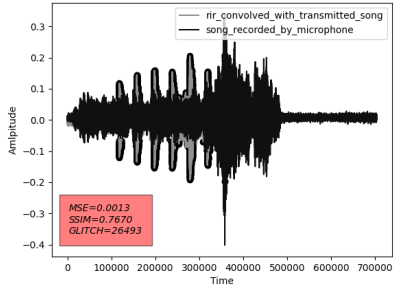
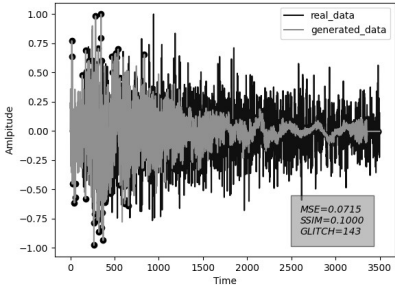
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model



0:00 / 0:43

6

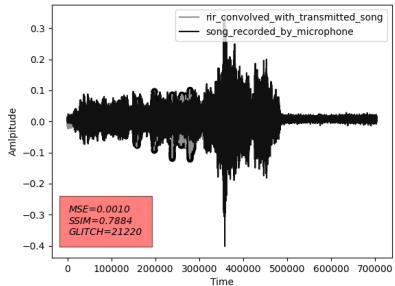
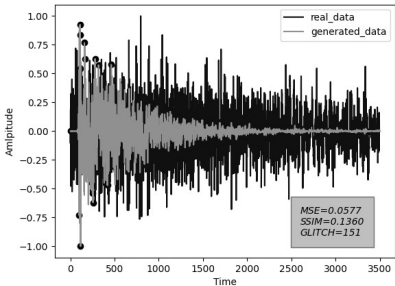
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model



0:00 / 0:43

8

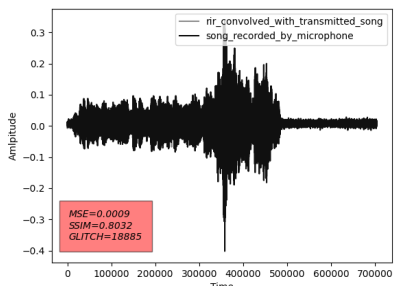
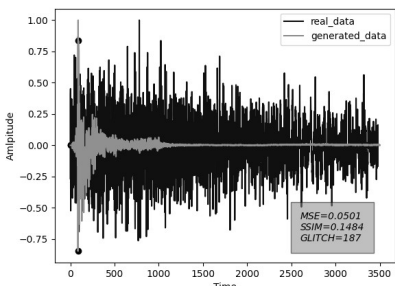
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:43

8

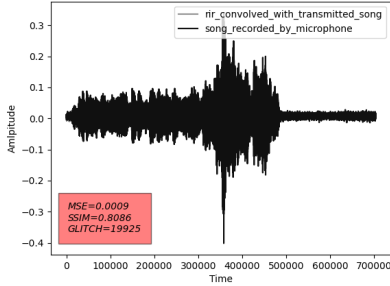
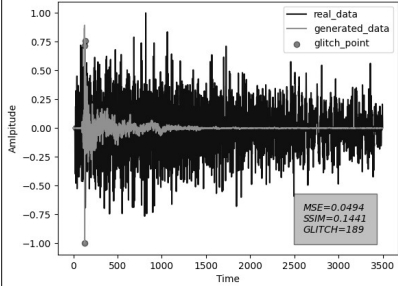
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:43

7

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



8

ROOM : room-z04
Room Dimensions - Width, Depth, Height (m) : 8.27x5.16x3.28
Total Room Volume (m³) : 140.0
Total Room Area (m²) : 173.4
Speaker Coordinates X,Y,Z (m) : 1.30x2.15x1.86
Microphone Coordinates X,Y,Z (m) : 3.98x5.31x2.07

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

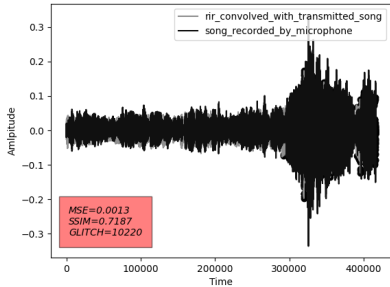
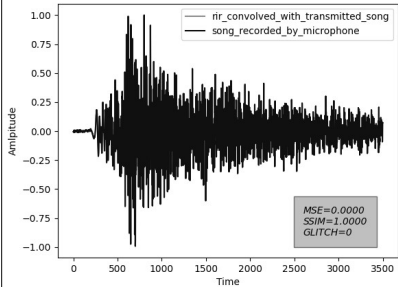
-

-



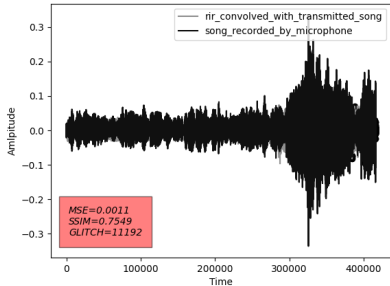
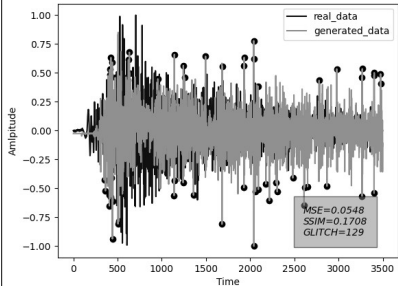
10

Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



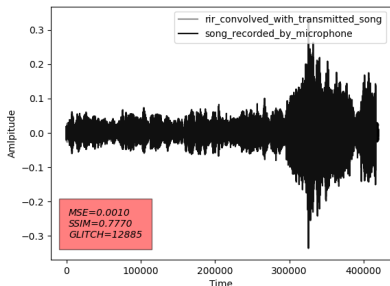
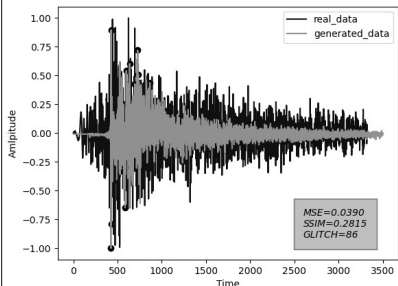
9

Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model

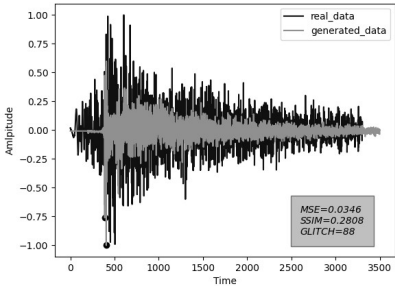
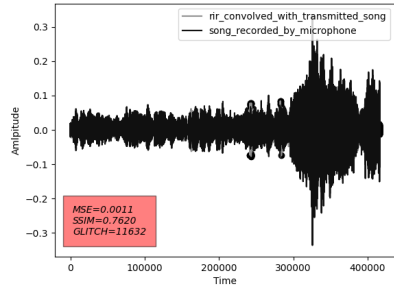
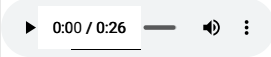
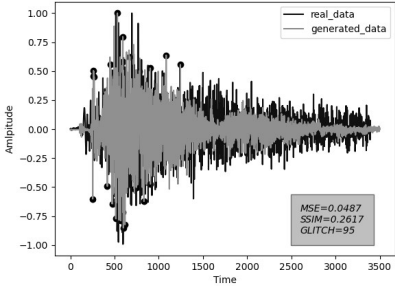
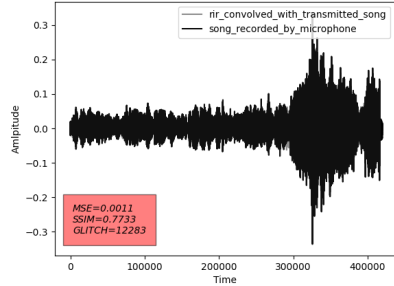
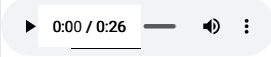
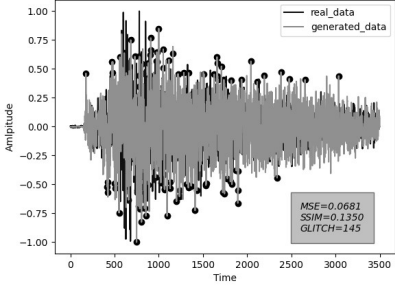
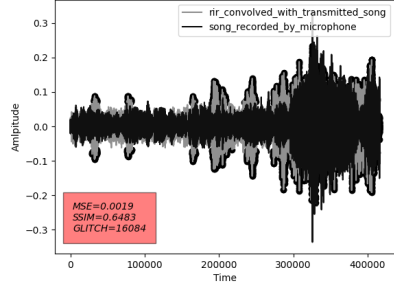
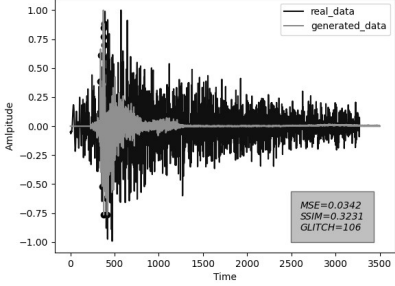
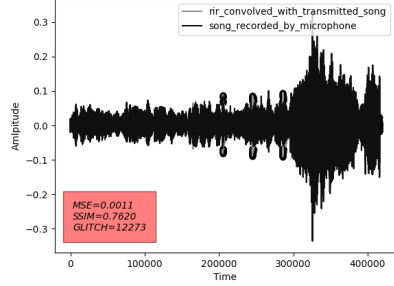
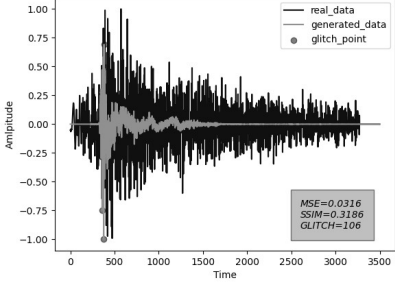
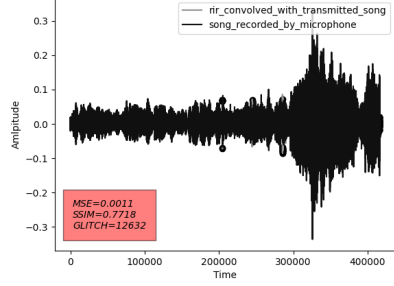
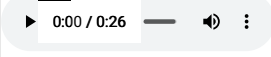


5

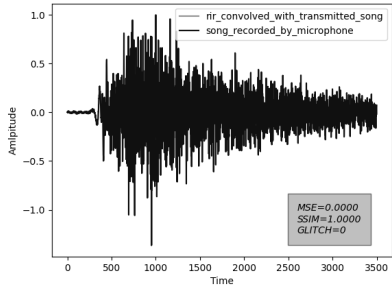
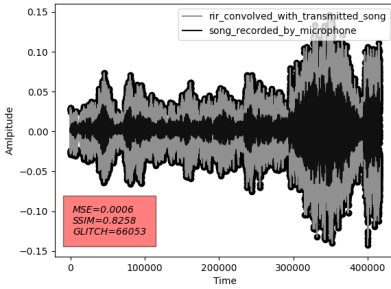
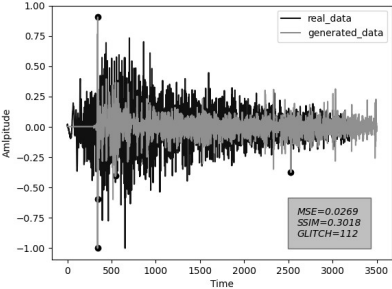
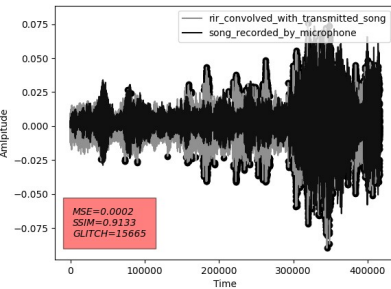
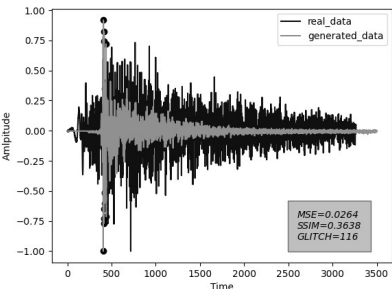
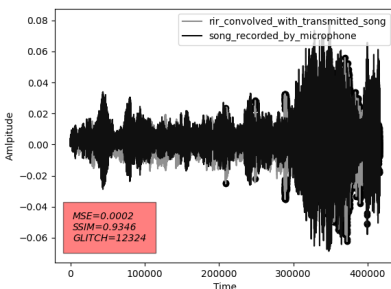
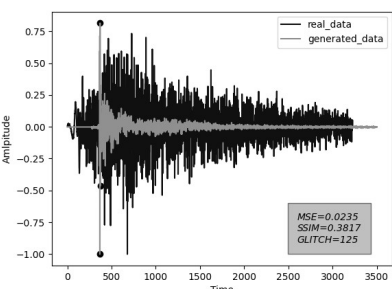
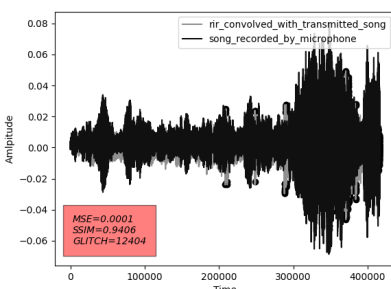
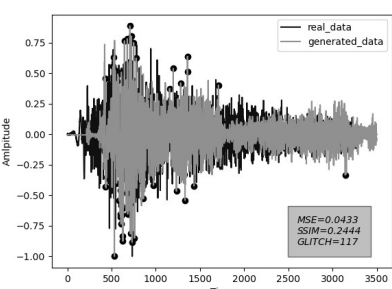
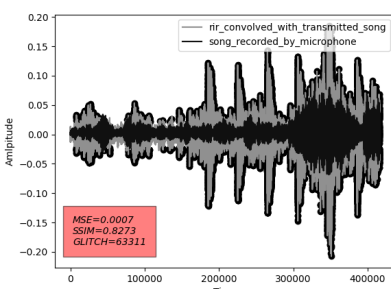
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



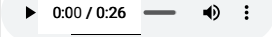
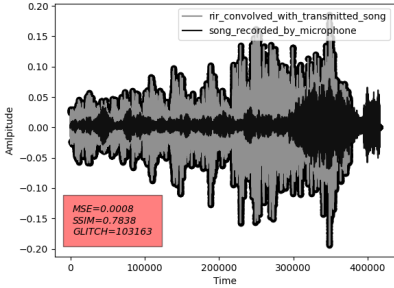
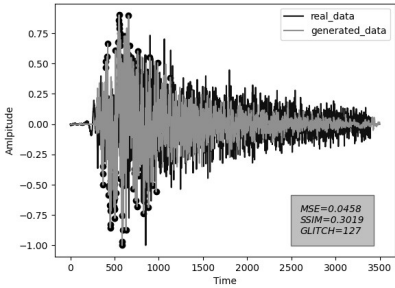
6

Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model				7
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model				8
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model				8
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model				9
Artificially Reverberated Song Using RIR Generated by MESHTAE Model				10

ROOM : room-z06
Room Dimensions - Width, Depth, Height (m) : 11.6x8.53x3.3
Total Room Volume (m³) : 326.5
Total Room Area (m²) : 330.8
Speaker Coordinates X,Y,Z (m) : 6.66x5.08x1.67
Microphone Coordinates X,Y,Z (m) : 3.27x5.65x2.60

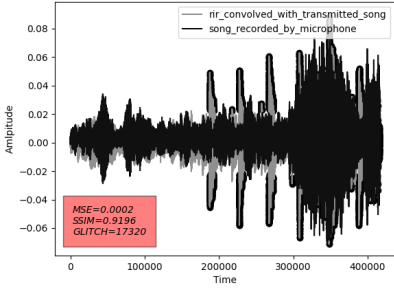
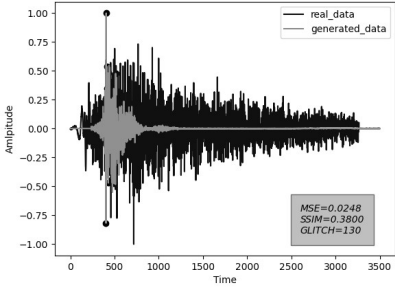
TYPE	RIR COHERENCE PLOT	SONG COHERENCE PLOT	AUDIO	RATING
Real Reverberated Song Recorded Using Microphone	-	-	▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal			▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	7
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model			▶ 0:00 / 0:26 — 🔊 ⋮	8
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	9
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model			▶ 0:00 / 0:26 — 🔊 ⋮	9

Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



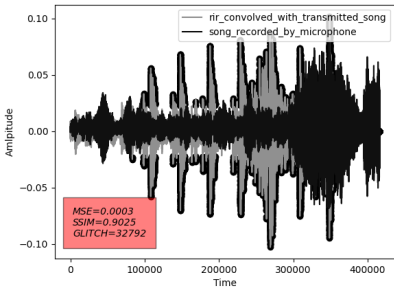
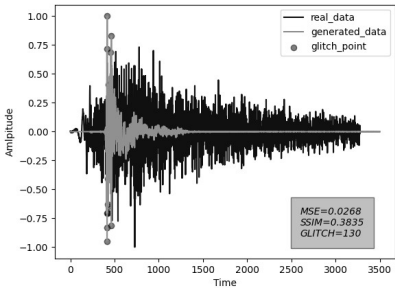
7

Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



8

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



7

ROOM : room-z10
Room Dimensions - Width, Depth, Height (m) : 11.84x8.54x3.25
Total Room Volume (m³) : 328.6
Total Room Area (m²) : 334.7
Speaker Coordinates X,Y,Z (m) : 5.08x10.09x1.89
Microphone Coordinates X,Y,Z (m) : 3.57x5.17x2.57

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

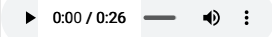
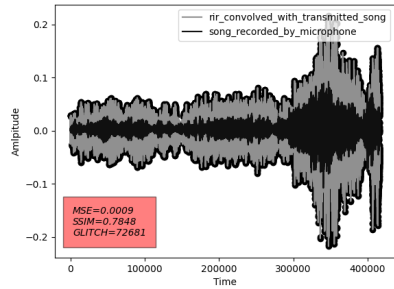
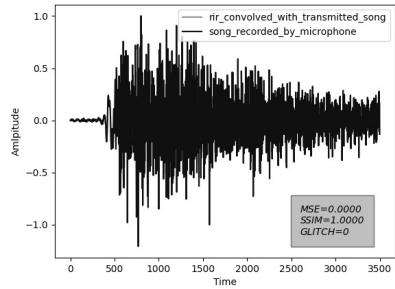
-

-



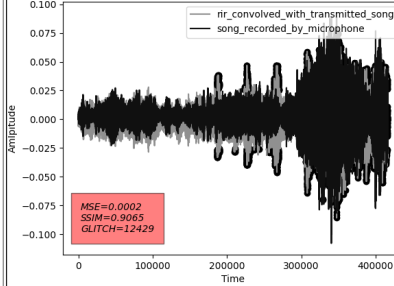
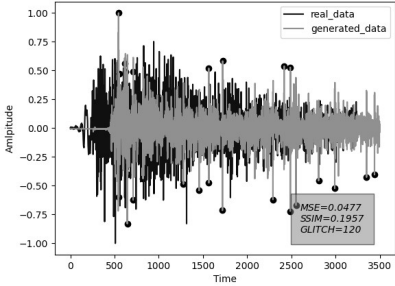
10

Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



7

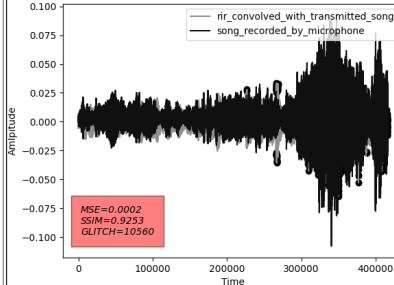
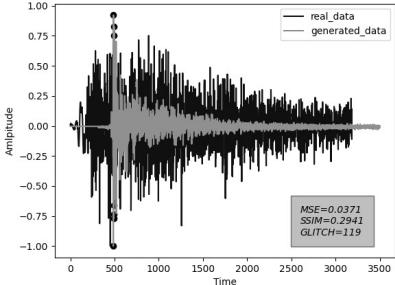
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model



0:00 / 0:26

8

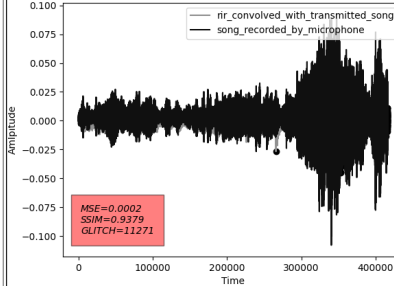
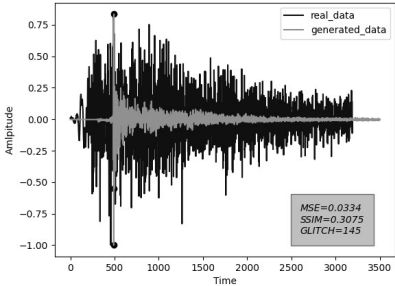
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



0:00 / 0:26

9

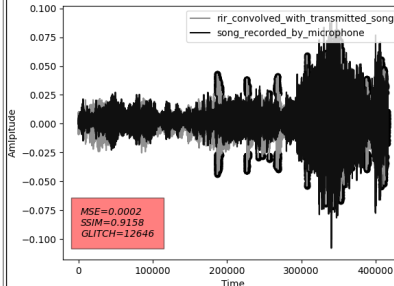
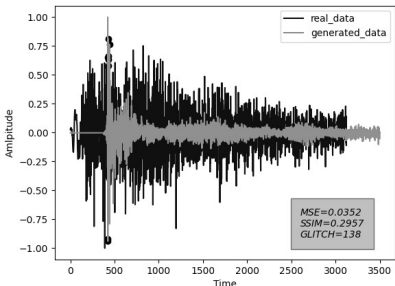
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model



0:00 / 0:26

9

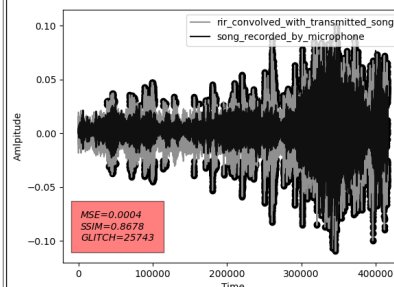
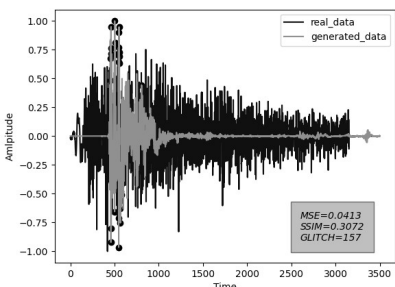
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model



0:00 / 0:26

8

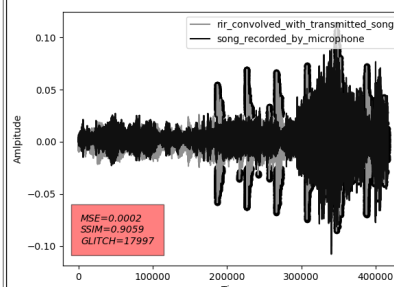
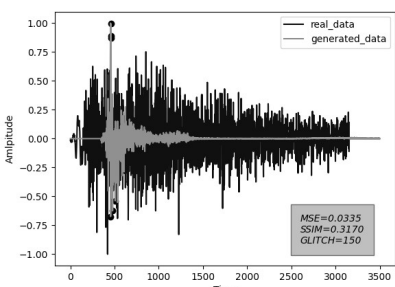
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:26

8

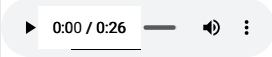
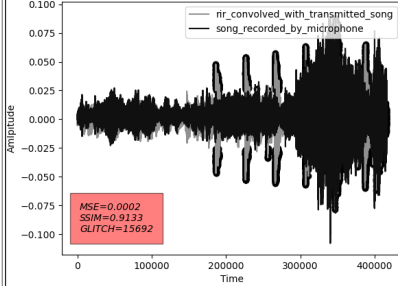
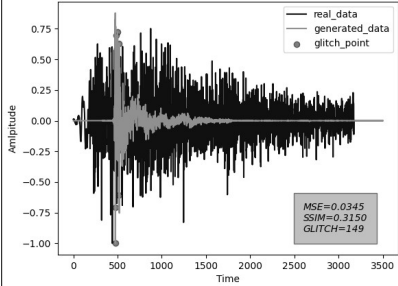
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:26

9

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



9

ROOM : room-z11
Room Dimensions - Width, Depth, Height (m) : 5.83x10.8x3.25
Total Room Volume (m^3) : 204.6
Total Room Area (m^2) : 234.0
Speaker Coordinates X,Y,Z (m) : 9.24x1.65x1.99
Microphone Coordinates X,Y,Z (m) : 4.47x3.57x2.57

TYPE

RIR COHERENCE PLOT

SONG COHERENCE PLOT

AUDIO

RATING

Real Reverberated Song Recorded Using Microphone

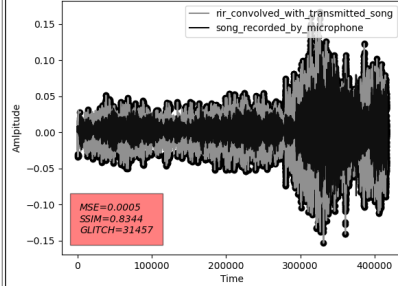
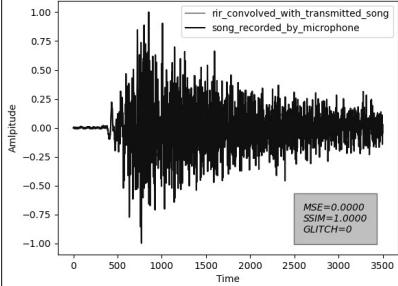
-

-



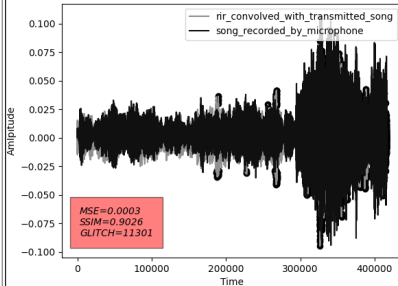
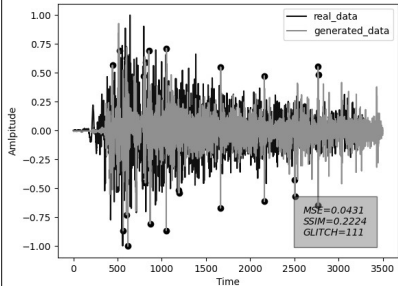
10

Artificially Reverberated Song Using Real RIR Measured Using ESS Signal



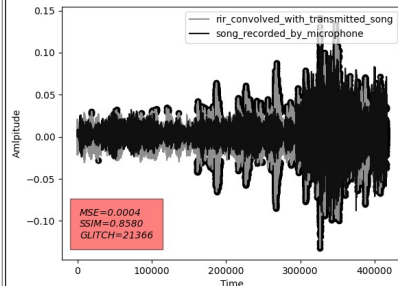
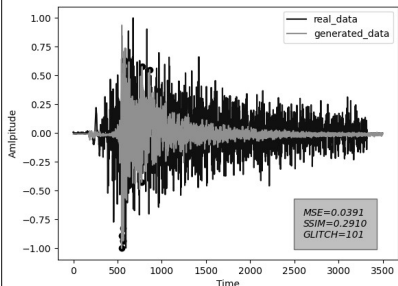
7

Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model



8

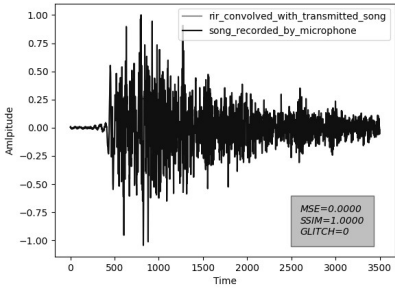
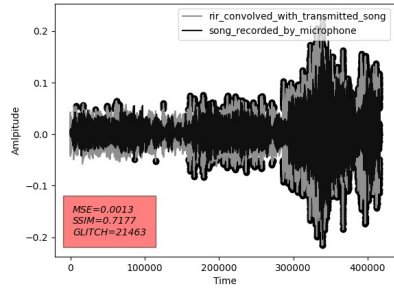
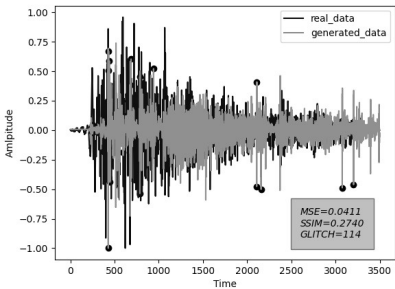
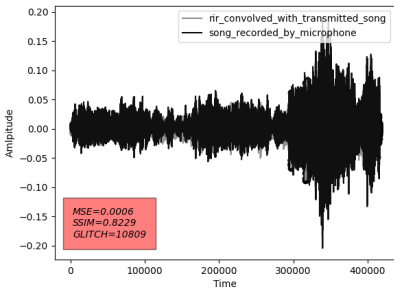
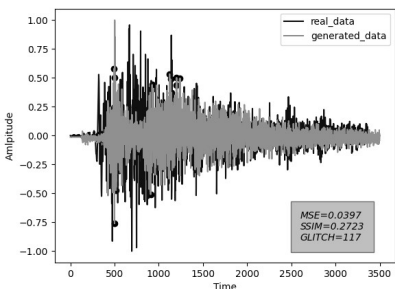
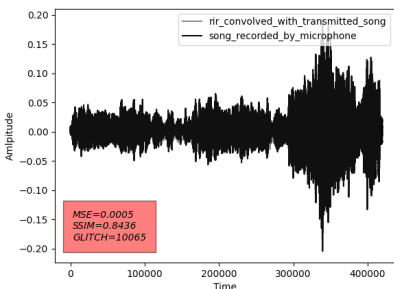
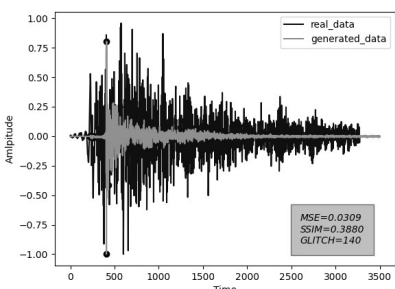
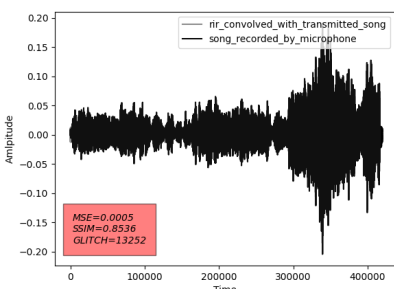
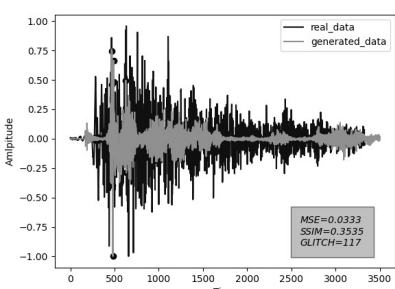
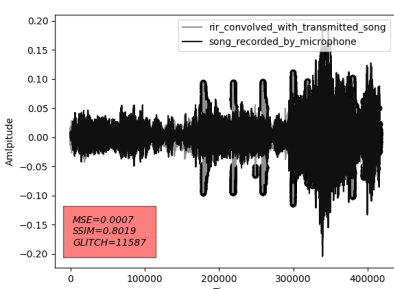
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model



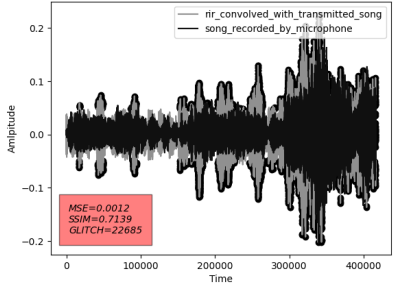
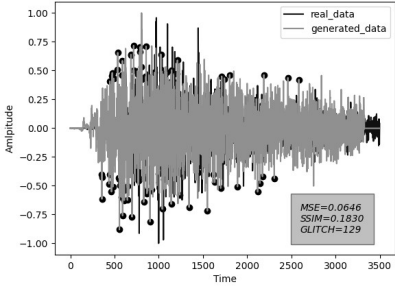
7

Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model	MSE=0.0333 SSIM=0.3059 GLITCH=133	MSE=0.0002 SSIM=0.9218 GLITCH=11708		7
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model	MSE=0.0697 SSIM=0.1978 GLITCH=184	MSE=0.0013 SSIM=0.7496 GLITCH=75626		8
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model	MSE=0.0630 SSIM=0.1880 GLITCH=115	MSE=0.0010 SSIM=0.7507 GLITCH=77911		9
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model	MSE=0.0336 SSIM=0.2940 GLITCH=125	MSE=0.0003 SSIM=0.8821 GLITCH=20842		8
Artificially Reverberated Song Using RIR Generated by MESHTAE Model	MSE=0.0333 SSIM=0.3164 GLITCH=137	MSE=0.0002 SSIM=0.9255 GLITCH=16873		7

ROOM : room-z23
Room Dimensions - Width, Depth, Height (m) : 11.5x12.3x2.9
Total Room Volume (m³) : 410.2
Total Room Area (m²) : 420.9
Speaker Coordinates X,Y,Z (m) : 10.07x7.01x1.86
Microphone Coordinates X,Y,Z (m) : 4.01x3.96x1.60

TYPE	RIR COHERENCE PLOT	SONG COHERENCE PLOT	AUDIO	RATING
Real Reverberated Song Recorded Using Microphone	-	-	▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using Real RIR Measured Using ESS Signal	 Amplitude vs Time (0-3500). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0000, SSIM=1.0000, GLITCH=0.	 Amplitude vs Time (0-40000). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0013, SSIM=0.7177, GLITCH=21463.	▶ 0:00 / 0:26 — 🔊 ⋮	9
Artificially Reverberated Song Using RIR Generated by FASTRIR-MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black), generated_data (grey). Metrics: MSE=0.0411, SSIM=0.2740, GLITCH=114.	 Amplitude vs Time (0-40000). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0006, SSIM=0.8229, GLITCH=10809.	▶ 0:00 / 0:26 — 🔊 ⋮	8
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM Model	 Amplitude vs Time (0-3500). Legend: real_data (black), generated_data (grey). Metrics: MSE=0.0397, SSIM=0.2723, GLITCH=117.	 Amplitude vs Time (0-40000). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0005, SSIM=0.8436, GLITCH=10065.	▶ 0:00 / 0:26 — 🔊 ⋮	9
Artificially Reverberated Song Using RIR Generated by FASTRIR-SSIM_PLUS_MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black), generated_data (grey). Metrics: MSE=0.0309, SSIM=0.3880, GLITCH=140.	 Amplitude vs Time (0-40000). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0005, SSIM=0.8536, GLITCH=13252.	▶ 0:00 / 0:26 — 🔊 ⋮	10
Artificially Reverberated Song Using RIR Generated by MESH2IR-MSE Model	 Amplitude vs Time (0-3500). Legend: real_data (black), generated_data (grey). Metrics: MSE=0.0333, SSIM=0.3535, GLITCH=117.	 Amplitude vs Time (0-40000). Legend: rir_convolved_with_transmitted_song (grey), song_recorded_by_microphone (black). Metrics: MSE=0.0007, SSIM=0.8019, GLITCH=11587.	▶ 0:00 / 0:26 — 🔊 ⋮	7

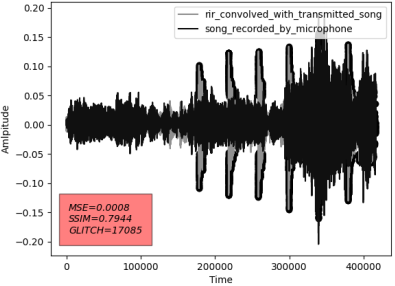
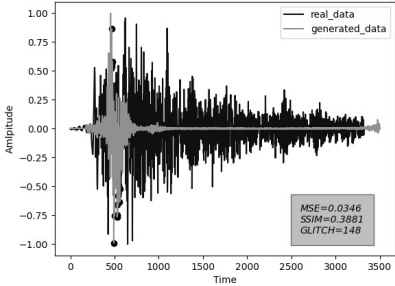
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM Model



0:00 / 0:26

8

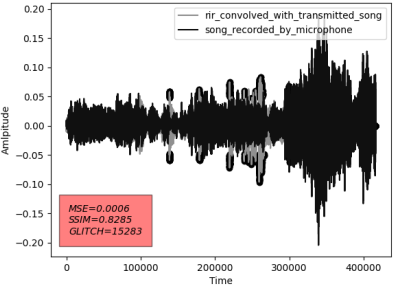
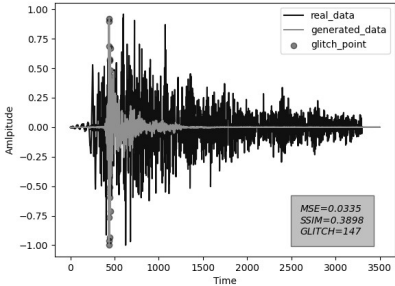
Artificially Reverberated Song Using RIR Generated by MESH2IR-SSIM_PLUS_MSE Model



0:00 / 0:26

9

Artificially Reverberated Song Using RIR Generated by MESHTAE Model



0:00 / 0:26

10