

Evaluation of Audio-Visual Virtual Reality Scenes: A 10-Week Technical Internship Report

Muhammad Hazimi Bin Yusri

June - August 2024

Abstract

This technical report details a 10-week internship focused on improving the evaluation of audio-visual virtual reality scenes. The project encompassed three main areas: refining a machine learning pipeline for scene reconstruction, implementing VR scene rendering in Unity with spatial audio, and developing acoustic evaluation methods using MATLAB. Key achievements include adapting EdgeNet360 for 360° image processing, creating a custom audio recording system in Unity, and addressing challenges in room impulse response measurements and acoustic parameter calculations.

Contents

1	Introduction	3
1.1	Background and Context	3
1.2	Project Objectives	3
1.3	Overview of the VR Audio Evaluation System	3
1.4	Scope of the Internship	3
2	Machine Learning Pipeline for Scene Reconstruction	4
2.1	Overview of the ML Pipeline	4
2.2	360monodepth Module	4
2.3	Material Recognition	5
2.4	EdgeNet360	5
3	Unity Implementation for VR Scene Rendering	6
3.1	Scene Import and Material Assignment	6
3.2	Steam Audio Integration	6
3.2.1	Room Configuration for RIR Simulation	6
3.3	Custom Audio Recording System	7
3.3.1	Direct Unity Audio Capture Implementation	7
3.3.2	Synchronization and Timing Considerations	7
3.4	Sine Sweep Generation and Playback	8
3.4.1	MATLAB Integration for Sweep Generation	8
3.4.2	Unity Playback and Recording Workflow	8
3.5	Scene Interaction and Navigation	8
3.6	Challenges Encountered and Solutions Implemented	8

3.7	Recommendations for Future Unity Development	9
4	Acoustic Evaluation using MATLAB	9
4.1	Room Impulse Response (RIR) Processing	9
4.1.1	Deconvolution Techniques	9
4.1.2	Noise Floor Analysis and Mitigation	9
4.2	Acoustic Parameter Calculation	10
4.2.1	EDT (Early Decay Time) Computation	10
4.2.2	RT60 (Reverberation Time) Estimation	11
4.2.3	Frequency Band Analysis	11
4.3	Data Visualization and Comparison Tools	11
4.4	Experimental Results across Different Scenes	12
4.4.1	Comparative Analysis with Previous Studies	12
4.4.2	Interpretation of Results	12
5	Conclusion and Future Work	13
5.1	Summary of Key Achievements	13
5.2	Evaluation of Project Objectives	13
5.3	Lessons Learned	14
5.4	Recommendations for Future Research and Development	14
A	Scene import table from Week 5	15
B	Audio Results	15
C	AV-VR University Provided Laptop: Technical Considerations	16
C.1	Project Repository and File Management	16
C.2	Matlab Tools and Modifications	16
C.3	Version Control and Change Tracking	16
C.4	Operating System and VR Hardware Compatibility	16

1 Introduction

1.1 Background and Context

Audio-visual virtual reality (VR) scene evaluation has become increasingly important in the field of immersive technologies. Recent advancements in this area have paved the way for more accurate and efficient methods of recreating indoor environments in VR. Notably, the work of Dr. Hansung Kim et al. (2021) achieved highly accurate audio and visual regeneration of indoor room scenes using stereo depth matching and 3D-CNN voxel mesh techniques, coupled with manual assignment of Steam Audio materials.

Building upon this foundation, subsequent Group Design Project (GDP) work focused on enhancing the process by utilizing a single image (monodepth) approach and automating the acoustic material assignment through the implementation of a Dynamic Backward Attention Transformer (DBAT). This progression is crucial for simplifying the capture process and improving the scalability of VR scene recreation.

However, the transition to EdgeNet360, which was built upon stereo depth map principles, resulted in mesh quality that did not meet the standards set by previous work. This setback necessitated further refinement of the mesh generation process and consequently affected the conclusiveness of the audio evaluation component.

1.2 Project Objectives

The primary objectives of this internship project were twofold:

1. To organize and consolidate the existing GDP work, with a focus on obtaining proper audio evaluation metrics (RT60 and EDT) for all reconstructed scenes.
2. To improve upon the monodepth scene generation process, addressing the limitations observed in the current implementation of EdgeNet360.

1.3 Overview of the VR Audio Evaluation System

The VR audio evaluation system comprises three main components:

1. **ML Pipeline:** Responsible for generating scene meshes from a single 360° image of an indoor room environment.
2. **Unity Project:** Incorporates automation features to streamline the process of scene rendering and audio simulation.
3. **MATLAB Scripts:** Utilized for evaluation purposes, including sine sweep generation, deconvolution, and octave band analysis to derive RT60 and EDT values. The EDT calculation script was provided by Dr. Atiyeh.

1.4 Scope of the Internship

This 10-week internship was designed to build upon and extend the previous GDP work where possible and applicable. The project was initially structured in two phases:

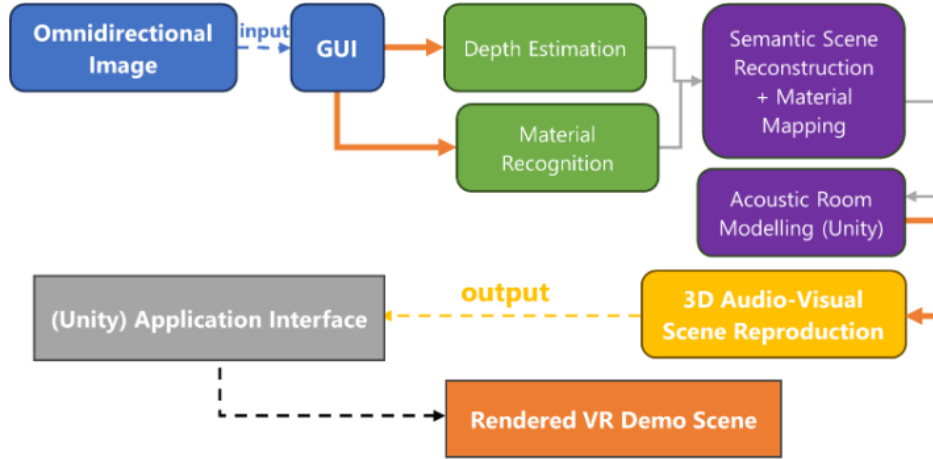


Figure 1: Pipeline flow diagram adapted from GDP’s report.

1. **First Half:** Focused on improving the existing GDP work and conducting thorough evaluations of the improvements.
2. **Second Half:** Originally planned to implement Mona’s code for further enhancements.

However, due to unforeseen challenges in the evaluation process, the scope of the second half was adjusted. The majority of the internship period was dedicated to refining the evaluation methodologies and addressing the setbacks encountered in the first phase, rather than proceeding with the implementation of additional features as initially planned.

2 Machine Learning Pipeline for Scene Reconstruction

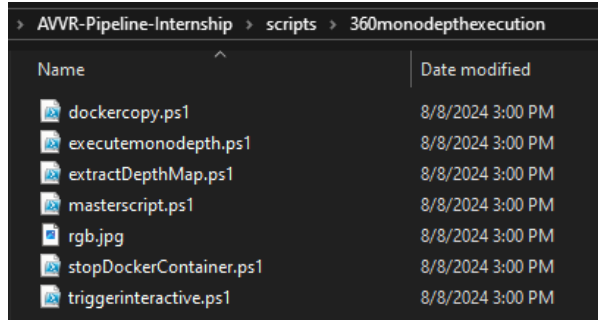
2.1 Overview of the ML Pipeline

The machine learning pipeline for scene reconstruction is a multi-stage process that has been automated using Python scripts. The primary script, `GUI.py`, orchestrates the execution of various modules through a `combined.bat` script. During this internship, the pipeline was modified to enhance modularity and adaptability across different systems. This was achieved by centralizing hardcoded directory paths at the beginning of the script.

For a comprehensive setup of the ML pipeline, users should refer to the `Manual.pdf` created by the GDP group, which outlines prerequisites and installation steps for all dependencies. It is important to note that while the GDP report contains more detailed information on each module, this section focuses primarily on modifications and findings from the internship period.

2.2 360monodepth Module

The 360monodepth module is executed within a Docker container to mitigate compatibility issues. A series of PowerShell scripts were developed to automate the process of



The screenshot shows a file explorer window with the path `> AVR-Pipeline-Internship > scripts > 360monodepthexecution`. It displays a list of files with their names and modification dates.

Name	Date modified
<code>dockercopy.ps1</code>	8/8/2024 3:00 PM
<code>executemonodepth.ps1</code>	8/8/2024 3:00 PM
<code>extractDepthMap.ps1</code>	8/8/2024 3:00 PM
<code>masterscript.ps1</code>	8/8/2024 3:00 PM
<code>rgb.jpg</code>	8/8/2024 3:00 PM
<code>stopDockerContainer.ps1</code>	8/8/2024 3:00 PM
<code>triggerinteractive.ps1</code>	8/8/2024 3:00 PM

Figure 2: PowerShell scripts for Docker automation and process orchestration

instantiating the Docker container, executing the module, and transferring input and output data to intermediary folders.

One persistent issue with this implementation is the accumulation of old Docker containers. While these do not significantly impact disk space, they clutter the Docker container list, which is not ideal from a system management perspective.

To maintain version control and facilitate future updates, the original 360monodepth repository was forked and is now maintained at <https://github.com/Muhammad-Hazimi-Yusri/360monodepth>. This forking strategy was applied to all submodules to ensure consistency and ease of management.

An investigation into monodepth boosting techniques was conducted, but the results indicated minimal improvement for our specific use case. Further details on this are discussed in the EdgeNet360 section.

2.3 Material Recognition

The material recognition component utilizes the Dynamic Backward Attention Transformer (DBAT) model. The original repository (<https://github.com/heng-yuwen/Dynamic-Backward-Attention-Transformer>) was forked and is now maintained at <https://github.com/Muhammad-Hazimi-Yusri/Dynamic-Backward-Attention-Transformer>.

Performance analysis of the DBAT model revealed generally satisfactory results across most scenes. However, anomalies were observed in specific scenarios, such as the Listening Room, where walls and ceilings were incorrectly classified as metal surfaces. This misclassification led to an unrealistically reverberant acoustic simulation. It is hypothesized that the lack of distinctive features in empty or minimalist scenes may contribute to these classification errors.

The impact of these misclassifications is potentially amplified by the GDP group’s decision to use a connected mesh instead of a voxel mesh, resulting in a loss of fine details. This trade-off between mesh connectivity and detail preservation warrants further investigation.

2.4 EdgeNet360

A significant issue identified in the EdgeNet360 module was the omission of the `enhance360.py` preprocessing step prior to executing `infer360.py`. This oversight resulted in the generation of floating meshes in the middle of the scene, particularly affecting ceiling and floor representations. The root cause was traced to improper normalization of the depth map for compatibility with EdgeNet360.

Implementation of the `enhance360.py` preprocessing step resolved the floating mesh issue but introduced a new challenge: loss of detail for near objects, particularly noticeable in smaller rooms like MR and UL containing furniture close to the camera position. This trade-off between global scene coherence and local detail preservation highlights the complexity of depth map preprocessing for 360° environments.

It remains unclear whether the GDP group intentionally omitted the enhancement step to preserve near-field details or if it was an oversight. Further investigation into this decision-making process is recommended.

A fundamental limitation of the current EdgeNet360 implementation is its foundation in stereo depth map principles. The monodepth maps generated by our pipeline exhibit smoother gradients and fewer artifacts compared to stereo depth maps. This discrepancy between the training data characteristics and the input data may contribute to the reduced detail in the final reconstructed scenes. Future work should focus on adapting EdgeNet360 to better accommodate the characteristics of monodepth input or exploring alternative network architectures specifically designed for monodepth-based reconstruction.

3 Unity Implementation for VR Scene Rendering

This section details the implementation of virtual reality (VR) scene rendering using Unity engine version 2022.3.11f1 LTS. It is worth noting that any version within the same major release should be compatible with the project structure.

3.1 Scene Import and Material Assignment

The Group Design Project (GDP) team developed a custom ribbon menu script (AVVR) to streamline the process of importing scenes and assigning Steam Audio mesh materials. This method proves more efficient than the standard drag-and-drop approach, as it includes the custom audio metadata crucial for accurate processing.

During the internship, modifications were made to enhance the modularity of the Unity project, particularly in making folder paths dynamic rather than hardcoded. This improvement allows for greater flexibility across different system configurations.

3.2 Steam Audio Integration

The project utilizes Steam Audio version 4.5.3. Initially, the GDP team opted for real-time processing to accommodate dynamic audio source positioning in VR using controller input. However, this approach necessitated lower accuracy settings to maintain real-time performance.

For the purposes of Room Impulse Response (RIR) evaluation, a shift towards baked reverb was implemented. This decision was based on the static nature of both the audio source and listener positions during evaluation. Additionally, Head-Related Transfer Function (HRTF) processing was disabled to eliminate unnecessary reverb effects dependent on camera orientation, which could potentially affect RIR measurements.

3.2.1 Room Configuration for RIR Simulation

To enhance the efficiency of scene setup, two prefabs were created:

- **DimensionCalculate:** This prefab assists in verifying the scale of imported scenes. However, it has limitations regarding axis orientation, necessitating manual inspection and pre-orientation to ensure correct length and width assignments.
- **AudioCameraPositioner:** This prefab facilitates the positioning of audio sources and the main camera (listener) in accordance with ground truth setups.

The following table outlines the dimensions and positions used for each scene:
[Insert table with scene dimensions and audio source/listener positions here]

3.3 Custom Audio Recording System

Concerns were raised regarding the fidelity of RIR recordings obtained through third-party tools such as OBS and Audacity. To address this, a custom direct internal audio capture system was implemented within Unity. While comparative testing showed minimal differences between Audacity recordings and the custom system, the latter offers several advantages:

- Elimination of Windows Volume Mixer dependencies, reducing the risk of audio clipping.
- Ability to run audio recordings while the system is muted, protecting audio hardware and the user’s hearing.
- Improved standardization of the recording process.

3.3.1 Direct Unity Audio Capture Implementation

The custom audio recording system comprises two primary scripts:

- **AudioCapture.cs:** Responsible for monitoring current audio.
- **AudioRecorded.cs:** Handles the recording and saving of .wav files.

These tools have been integrated into a custom window accessible via Window/Audio Recorder in the Unity top bar. The system was further refined to automatically initiate recording upon scene play, eliminating the need for manual activation.

3.3.2 Synchronization and Timing Considerations

While theoretically, precise synchronization should not significantly impact RIR analysis due to the frequency domain deconvolution process, additional timing controls were implemented as a precautionary measure. These include:

- A 2-second delay before audio source playback to ensure proper initialization of Steam Audio reverb effects.
- Disabling of autoplay and loop features on the audio source to allow for reverb tail capture.

Further validation of these timing considerations is recommended in future work.

3.4 Sine Sweep Generation and Playback

3.4.1 MATLAB Integration for Sweep Generation

Multiple versions of sine sweeps were tested, with the final implementation using `matlab_sweep` generated by `main1_mhby1g21.m`. Contrary to real-world scenarios where longer sweeps are preferred for improved signal-to-noise ratios, simulations showed better results with shorter sweeps (10 seconds). This phenomenon may be attributed to accumulated floating-point precision errors in longer simulations, contributing to noise floor issues.

3.4.2 Unity Playback and Recording Workflow

The workflow for playback and recording in Unity involves several critical steps:

1. Export probe box and Steam Audio active scene.
2. Bake probes for pathing, audio source, and listener reverb.
3. Configure desired file name and recording duration.
4. Ensure 'spatialise' is checked and 'spatial blend' is set to 3D in audio source settings.
5. Configure the steam audio source setting.
6. Make sure both Unity and Steam Audio listener, and baked listener component is added to correct object (most likely main camera).
7. Run the scene to initiate recording.
8. Save the .wav file in the designated location.

It is important to note that occasionally, the baked implementation may fail to activate Steam Audio, resulting in mono audio output. This issue requires further investigation, as it significantly impacts the accuracy of RIR measurements.

3.5 Scene Interaction and Navigation

Non-VR demo scenes with capsule colliders were added to facilitate quick testing in 2D environments. For VR interactions, minor adjustments were made to locomotion controls, adhering to standard VR experience norms. The grabable audio source feature requires further development and testing.

3.6 Challenges Encountered and Solutions Implemented

Several technical challenges were encountered during the Unity development process:

- **Version Control System (VCS) limitations:** The size of certain .obj files, particularly LiDAR data, exceeded Git LFS quotas. Alternative file sharing methods are recommended for large assets.
- **Unity coordinate system complexity:** A comprehensive reference table was created to address confusion regarding scene orientations and dimensions.

- **Project stability issues:** Occasional crashes were observed during filesystem operations and commits.
- **TrueAudioNext compatibility:** Attempts to use TrueAudioNext in global Steam Audio settings resulted in device-not-found errors, necessitating the use of standard convolution methods.

3.7 Recommendations for Future Unity Development

For future development of this project, the following recommendations are proposed:

- Initiate a new project with improved organization, referencing previous work only as needed.
- Develop a comprehensive test project for all Steam Audio features to ensure functionality and deepen understanding.
- Explore alternative audio plugins or game engines (e.g., Unreal Engine) that may offer superior documentation, examples, or focus on realism and research applications.

4 Acoustic Evaluation using MATLAB

4.1 Room Impulse Response (RIR) Processing

4.1.1 Deconvolution Techniques

Deconvolution is a critical process in obtaining the Room Impulse Response (RIR) from a recorded sine sweep. Mathematically, deconvolution can be expressed as:

$$h(t) = y(t) * x^{-1}(t) \quad (1)$$

where $h(t)$ is the RIR, $y(t)$ is the recorded output signal, $x^{-1}(t)$ is the inverse filter of the input sine sweep, and $*$ denotes convolution.

In practice, this process is typically performed in the frequency domain to improve computational efficiency:

$$H(f) = \frac{Y(f)}{X(f)} \quad (2)$$

where $H(f)$, $Y(f)$, and $X(f)$ are the Fourier transforms of $h(t)$, $y(t)$, and $x(t)$ respectively.

The inverse filter $x^{-1}(t)$ is designed to compensate for the frequency-dependent energy of the sine sweep, ensuring a flat frequency response in the resulting RIR.

4.1.2 Noise Floor Analysis and Mitigation

A persistent challenge throughout the internship was the presence of an unexpectedly high noise floor, particularly evident in higher octave bands. This issue significantly impacted the accuracy of the line of best fit used in acoustic parameter calculations.

After extensive testing and experimentation, an optimal configuration was achieved using a 10-second `matlab_sweep` and its corresponding inverse filter. This setup, preserved in the Eval scenes folder, yielded the lowest noise floor levels.

One hypothesis for this phenomenon is the accumulation of floating-point precision errors, which could explain why longer sweeps resulted in higher noise levels. This theory aligns with observations in digital signal processing where extended computations can lead to increased numerical errors.

To mitigate this issue, several approaches were considered:

- Windowing the RIR to focus on the most significant part of the response
- Applying noise reduction techniques such as spectral subtraction
- Implementing adaptive thresholding to distinguish between the actual reverb tail and the noise floor

Further investigation into the root cause of this noise floor issue is recommended for future work, potentially exploring alternative numerical precision methods or signal processing techniques.

4.2 Acoustic Parameter Calculation

4.2.1 EDT (Early Decay Time) Computation

Early Decay Time (EDT) is defined as six times the time taken for the sound energy to decay by 10 dB after the direct sound. It is particularly sensitive to early reflections and is often considered more correlated with the perceived reverberance of a space than RT60.

The EDT is calculated using a linear regression on the energy decay curve from 0 dB to -10 dB. Mathematically, this can be expressed as:

$$EDT = -6 \cdot \frac{10}{\text{slope}} \quad (3)$$

where the slope is determined from the best-fit line of the decay curve between 0 dB and -10 dB.

An unusual observation across most scenes was that EDT values consistently exceeded RT60 values. While not theoretically impossible, this is atypical, especially considering that ground truth measurements exhibited lower EDT values. This anomaly suggests that the simulated rooms have stronger early reflections that persist longer than expected, rather than decaying exponentially as typically observed in real spaces.

Potential causes for this phenomenon could include:

- Overemphasis of early reflections in the Steam Audio simulation
- Inaccuracies in the geometric modeling of the virtual spaces
- Limitations in the material properties assigned to surfaces

Further investigation into the Steam Audio simulation parameters and the acoustic modeling process is recommended to address this discrepancy.

4.2.2 RT60 (Reverberation Time) Estimation

Reverberation Time (RT60) is defined as the time taken for the sound energy to decay by 60 dB after the cessation of the sound source. Due to limitations in signal-to-noise ratio, RT60 is often estimated from a smaller decay range and extrapolated.

In this study, RT60 was estimated using the RT30 method, which measures the time for a 30 dB decay and extrapolates to 60 dB. This was implemented using the `y_fit` parameter set to [-5 -35] when running the analysis code. The calculation can be expressed as:

$$RT60 = 2 \cdot RT30 = -2 \cdot \frac{60}{\text{slope}} \quad (4)$$

where the slope is determined from the best-fit line of the decay curve between -5 dB and -35 dB.

This approach helps mitigate the impact of the noise floor on RT60 estimation by focusing on a more reliable portion of the decay curve.

4.2.3 Frequency Band Analysis

A significant challenge encountered in the frequency band analysis was the uneven distribution of RT60 values across octave bands, despite average values closely matching ground truth. This imbalance was particularly evident in the analysis of sound samples provided by Mona and was exacerbated by the aforementioned noise floor issues.

To address this, the `y_fit` parameter adjustment mentioned in the RT60 estimation was implemented. After consultation with Dr. Atiyeh, this was determined to be the most appropriate course of action, given that the root cause of the issue remains unresolved.

The importance of balanced frequency distribution cannot be overstated, as it serves as a crucial sanity check to ensure that accurate average RT60 values are not merely coincidental but rather the result of proper analysis across all frequency bands.

Future work should focus on:

- Investigating the source of frequency-dependent anomalies in the simulated RIRs
- Developing more robust methods for frequency band analysis that are less susceptible to noise floor issues
- Comparing frequency-dependent behavior between simulated and measured RIRs to identify potential improvements in the acoustic simulation process

4.3 Data Visualization and Comparison Tools

The primary MATLAB toolbox and modules used for data visualization and analysis were provided by Mona, based on Dr. Atiyeh's work. These tools offered comprehensive visualization capabilities for individual RIR analyses.

To facilitate comparison with previous results, additional custom graphing scripts were developed. These scripts allowed for:

- Side-by-side comparison of RT60 and EDT values across different methods (e.g., monodepth vs. stereo depth)

- Visualization of frequency-dependent acoustic parameters for multiple scenes simultaneously
- Statistical analysis of differences between simulated and ground truth values

Future enhancements to these tools could include automated report generation and interactive visualization interfaces to streamline the analysis process.

4.4 Experimental Results across Different Scenes

4.4.1 Comparative Analysis with Previous Studies

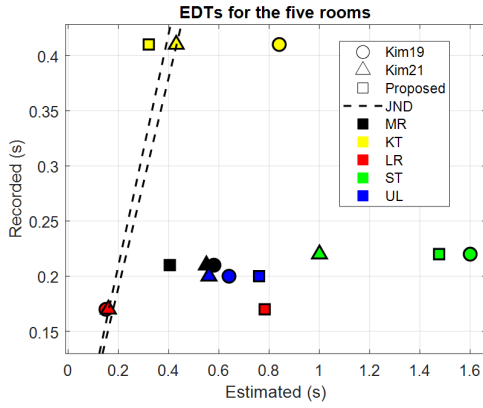


Figure 3: Early Decay Times for the five rooms, related to the estimated RIRs in VR environment. The dashed lines show the JND limit of 5% (Vorländer 1995).

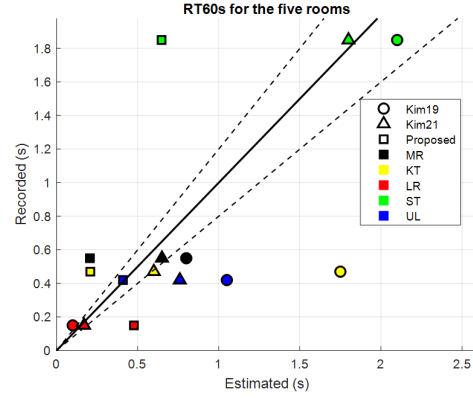


Figure 4: Reverberation Times (60dB) for the five rooms, related to the estimated RIRs in VR environment. The dashed lines show the JND limit of 20% (Meng et al. 2006).

4.4.2 Interpretation of Results

The acoustic evaluation results revealed several interesting patterns:

- Decay curves for almost all scenes exhibited a less exponential nature than expected, resulting in higher EDT values compared to RT60. This suggests a potential overemphasis on early reflections in the simulation process.
- Despite these anomalies, the overall trend in reverb behavior across different room sizes was generally consistent with expectations (i.e., larger rooms exhibited longer reverberation times).
- The accuracy of RT60 averages was relatively high, particularly for scenes reconstructed using monodepth techniques. This is somewhat surprising given that the room scale was manually adjusted post-import, which might have been expected to yield better results with stereo depth baselines.
- Material assignment played a crucial role in acoustic accuracy. The Listening Room (LR) scene, which had incorrectly assigned materials, showed significant deviations from expected results, underscoring the importance of accurate material properties in acoustic simulations.

While these results demonstrate the potential of the current approach, several areas for improvement were identified:

- Further investigation into the causes of non-exponential decay curves, potentially through more detailed analysis of early reflection patterns in the simulated RIRs.
- Refinement of material assignment techniques to improve acoustic accuracy, possibly through the development of more sophisticated automatic material recognition algorithms.
- Exploration of advanced calibration techniques to better align simulated acoustic properties with ground truth measurements across all frequency bands.

In conclusion, while the current implementation shows promise in reproducing general acoustic trends, particularly in terms of average RT60 values, there is significant room for improvement in accurately simulating the fine details of room acoustics, especially in the early reflection phase and across different frequency bands.

5 Conclusion and Future Work

5.1 Summary of Key Achievements

This 10-week internship has yielded significant technical accomplishments in the field of audio-visual virtual reality (VR) scene evaluation. The machine learning pipeline for scene reconstruction was enhanced, particularly in the areas of 360° monodepth estimation and EdgeNet360 implementation, resulting in improved modularity and efficiency. In the Unity environment, a more streamlined process for importing and rendering VR scenes was developed, including the creation of custom prefabs for dimension calculation and audio-camera positioning. A notable achievement was the implementation of a direct Unity audio capture system, which eliminated the need for external recording software and improved the standardization of the recording process. The MATLAB-based acoustic evaluation process was refined, with particular focus on addressing noise floor issues and improving the estimation of RT60 and EDT across different frequency bands. Additionally, the project’s modularity was improved to enhance compatibility across different system configurations, facilitating easier collaboration and development.

5.2 Evaluation of Project Objectives

The primary objectives of the internship were largely met, though with varying degrees of success across different areas. Improvements were made to the monodepth scene generation process, including the integration of `enhance360.py` preprocessing, which helped resolve floating mesh issues. However, challenges remain in achieving the level of detail observed in previous stereo depth approaches, particularly in preserving near-field object details.

Significant progress was made in refining the acoustic evaluation process, particularly in addressing noise floor issues. The development of custom Unity tools and the refinement of the ML pipeline significantly improved the overall workflow efficiency. However, the persistent anomaly of higher EDT values compared to RT60 across most scenes indicates that further refinement is needed in the simulation of early reflections. While

average RT60 values showed promising accuracy, particularly for monodepth reconstructions, inconsistencies in frequency band distribution and early reflection behavior suggest that additional work is required to achieve consistent accuracy across all acoustic parameters and scenes.

Version control system limitations for large assets remain a challenge to be addressed, highlighting the need for more robust solutions in collaborative VR development environments.

5.3 Lessons Learned

The internship provided several key technical insights. The critical role of depth map preprocessing in 3D scene reconstruction was highlighted, particularly in the trade-off between global scene coherence and local detail preservation. The challenges encountered in accurately simulating early reflections and maintaining consistent acoustic behavior across frequency bands underscored the complexity of virtual acoustic environments.

An important hypothesis emerged regarding the potential contribution of accumulated floating-point precision errors to noise floor issues in longer audio simulations, presenting a crucial consideration for future audio processing implementations. The significant impact of material assignment on acoustic simulation accuracy, as evidenced by the Listening Room scene results, emphasized the need for robust material recognition and assignment techniques.

Throughout the project, the importance of streamlined workflows and modular design in VR development was reinforced, particularly in the context of multi-disciplinary projects involving machine learning, Unity development, and acoustic analysis. These insights provide valuable guidance for future work in this field.

5.4 Recommendations for Future Research and Development

Based on the experiences and insights gained during this internship, several areas are recommended for future research and development. The EdgeNet360 architecture could be modified to better accommodate the characteristics of monodepth input, potentially through targeted training on monodepth-derived datasets. This could help bridge the gap between the current performance and that achieved with stereo depth approaches.

The development of more sophisticated automatic material recognition algorithms is crucial. These could leverage multi-modal data, including visual, acoustic, and semantic information, to improve the accuracy of acoustic material assignment. This would address the issues observed in scenes like the Listening Room, where incorrect material assignments significantly impacted acoustic simulation accuracy.

Further investigation into early reflection simulation is necessary. An in-depth analysis of early reflection patterns in simulated RIRs could help identify and address the causes of non-exponential decay curves and EDT anomalies. This research could lead to more accurate representations of complex acoustic environments in VR settings.

A comprehensive comparison of various spatial audio plugins and game engines (e.g., Steam Audio vs. Google Resonance, Unity vs. Unreal Engine) would be valuable in identifying the most suitable tools for research-oriented VR acoustic simulations. This could help address some of the limitations encountered with the current Steam Audio implementation.

In conclusion, while significant progress has been made in improving the audio-visual VR scene evaluation pipeline, there remain exciting opportunities for further research and development. The insights gained from this internship provide a solid foundation for future work aimed at enhancing the realism and accuracy of virtual acoustic environments in VR applications. By addressing the challenges identified and building upon the achievements made, future research can continue to push the boundaries of immersive audio-visual experiences in virtual reality.

References

A Scene import table from Week 5

Dimension (x, y, z) of scene meshes from original `enhance360.py` and `360monodepth` to calculate scale ratio needed as follows:

1. KT – (4.16, 2.08, 5.44) enter scale (0.83, 1.28, 1.22), align corner 2,
2. MR – (4.64, 1.84, 5.28), first rotate -90, enter scale (1.06, 1.27, 0.92), align corner 3
3. LR – (5.12, 2.08, 5.60), enter scale in table, align corner 0, swap X and Z for coords place.
4. ST – (6.08, 2.08, 6.40), first rotate -90, enter scale (2.67, 3.125, 2.39)
5. UL – (4.48, 1.92, 4.24), enter scale in table, align corner 0

Scene and ratio (x, y, z)	Cam/Listener coords	Audio/Source coords	Ground Truth Dimen
KT (0.83, 1.28, 1.22)	(4.100, 1.705, 1.585)	(1.953, 1.676, 1.982)	(3.46, 2.67, 6.64)
MR (0.92, 1.27, 1.06)	(2.12, 1.00, 0.33)	(2.12, 1.00, 3.00)	(4.28, 2.33, 5.61)
LR (0.99, 1.39, 1.01)	(2.55, 1.08, 2.79)	(0.51, 1.20, 2.80)	(5.05, 2.90, 5.64)
ST (2.39, 3.125, 2.67)	(6.94, 1.50, 5.00)	(4.94, 1.50, 5.00)	(14.55, 6.50, 17.08)
UL (1.16, 1.52, 1.31)	(3.58, 1.07, 2.27)	(1.32, 1.07, 2.24)	(5.20, 2.91, 5.57)

Table 2 Comparison of reconstruction errors in room dimensions

Data	Ground truth	Kim19 (Kim et al. 2019)		Proposed	
	Dimension (m)	Dimension (m)	Err in Dim (%)	Dimension (m)	Err in Dim (%)
MR	5.61 × 4.28 × 2.33	5.52 × 4.35 × 2.36	(1.60, 1.64, 1.29)	5.54 × 4.24 × 2.40	(1.25 , 0.93 , 3.00)
KT	6.64 × 3.46 × 2.67	6.95 × 3.41 × 2.70	(4.67, 1.45 , 1.12)	6.42 × 3.52 × 2.68	(3.31 , 1.73, 0.37)
LR	5.64 × 5.05 × 2.90	5.77 × 5.17 × 2.98	(2.30 , 2.38, 2.76)	5.88 × 5.02 × 2.78	(4.26, 0.59 , 4.14)
ST	17.08 × 14.55 × 6.50	16.53 × 14.87 × 5.70	(3.22, 2.20 , 12.31)	17.54 × 15.46 × 5.56	(2.69 , 6.25, 14.46)
UL	5.57 × 5.20 × 2.91	5.92 × 4.95 × 2.95	(6.28, 4.81, 1.37)	5.52 × 5.22 × 3.00	(0.90 , 0.38 , 3.09)

The bold figures represent the minimum error in each dimension

From <https://doi.org/10.1007/s10055-021-00594-3>, in Z, X, Y (Length, Width, Height).

Figure 5: This includes the audio source, listener positions and the ground truth of scene’s dimensions. Open Intern-Logs/Week 5.docx, Wednesday, Dimensions subheading to access original table for easier copy/paste.

B Audio Results

Note: It is possible to achieve higher RT60 and EDT values; however, this would also increase the values for other scenes. Fine-tuning for each scene individually by changing the global Steam Audio setting is not considered ideal or scientific for this comparative study. For Kim21 and Kim19, the values are approximated from the graph in Kim21 paper.

Table 1: Acoustic Results Comparison using Default Baked Settings

Scene	Proposed		Octave Bands (s)				
	RT60 (s)	EDT (s)	500 Hz	1 kHz	2 kHz	4 kHz	8 kHz
MR	0.2061	0.4045	0.19	0.21	0.20	0.21	0.21
KT	0.2084	0.3208	0.18	0.22	0.22	0.22	0.23
LR	0.4773	0.7821	0.50	0.50	0.49	0.48	0.45
ST	0.6477	1.4762	0.72	0.62	0.64	0.60	0.62
UL	0.4098	0.7595	0.44	0.43	0.43	0.38	0.47

C AV-VR University Provided Laptop: Technical Considerations

C.1 Project Repository and File Management

The majority of the work completed during this internship has been uploaded to the project repository. However, due to size constraints, certain large files have been excluded from the repository and are stored locally as seen in .gitignore file. These include:

- LiDAR scene of the Kitchen
- Original Unity folder (.zip files)
- Probe batch assets (.asset files), particularly for the Studio (ST) scene with high probe counts

These files can be found locally at `C:/Project/AVVR-Pipeline-Internship`. For archival purposes, the original 2023/24 GDP project folder remains untouched at `C:/Project/AV-VR`.

C.2 Matlab Tools and Modifications

The Atiyeh-RIR-Evaluation-Matlab tools, initially provided by Mona, are preserved in their original state within the base Project folder directory. Any modifications made to these tools have been implemented in separate copies to maintain the integrity of the original files.

C.3 Version Control and Change Tracking

For a detailed record of individual changes and development progress, refer to the commit history in the Git repository. The corresponding GitHub repository can be accessed at: <https://github.com/Muhammad-Hazimi-Yusri/AVVR-Pipeline-Internship>

C.4 Operating System and VR Hardware Compatibility

The university-provided laptop is running Windows 11, which introduces some considerations for VR development:

- The HP Reverb G2 VR headset used in this project requires Windows Mixed Reality support.

- Microsoft has announced plans to deprecate Windows Mixed Reality in future updates.
- As a university-managed device, automatic updates cannot be disabled without administrative intervention.

This situation presents a potential risk to the long-term compatibility of the current VR setup with the laptop. Future users of this project should be aware that system updates might impact the functionality of the VR headset, potentially requiring alternative VR solutions or administrative action to maintain compatibility.