Diegetic Sound Visualization for Hard-of-Hearing Users

Martin Nastoupil *
Supervised by: Mgr. Jiří Chmelík, Ph.D., †and RNDr. Vít Rusňák, Ph.D.‡

Faculty of Informatics
Masaryk University
Czech Republic / Brno

Abstract

Spatial audio is a crucial component of modern video games. However, current accessibility solutions for Deaf and Hard-of-Hearing (DHH) players often rely on 2D screen-space overlays that can break immersion and lack precise spatial context. This paper presents and explores a diegetic approach to sound visualization, proposing a shift from traditional interface overlays to in-world representations. By using real-time ray casting to approximate physical sound propagation, the system generates visual cues at locations where sound waves interact with game geometry. The resulting prototype offers a highly integrable, performance-conscious framework for developers, providing an immersive alternative for spatially accurate accessibility features in interactive media.

Keywords: Audio-visualization, Deaf or Hard of Hearing, Diegetic

1 Introduction

In modern video games, audio is far more than just an immersive aesthetic. It is a critical channel for conveying *spatial acoustics* and real-time gameplay feedback. However, this heavy reliance on sound creates a significant accessibility barrier for players who are Deaf or Hard of Hearing (DHH) [2, 3]. Unfortunately, this disadvantage affects a substantial and growing audience. According to the World Health Organization, over 5% of the global population (430 million people) currently live with disabling hearing loss, a figure projected to exceed 700 million by 2050, making it the third leading disability in the world [10].

While existing solutions, such as subtitles and captions, effectively convey dialogue, they often fail to convey the *spatio-temporal data* required for accurate reactive gameplay [5]. In First-Person Shooter (FPS) games, for instance, *acoustic events* such as footsteps or gunfire provide vital positional cues transmitted exclusively through audio. Without access to this data, DHH players experience a *spatial information deficit*, which diminishes their overall experience and underscores the urgent need for robust, inclusive game design [2].

To mitigate this gap, researchers have explored various visual alternatives to spatial audio. While audio-visual cues can enhance the subjective experience, they may be implemented with care to avoid sensory overload [6, 5]. Notably, Granados [4] observed that Hard of Hearing (HOH) participants occasionally preferred a "no cue" condition over the accessibility features. This preference likely stems from the design of these features; the study's qualitative feedback indicated that participants found the cue's binary nature insufficient for complex tasks.

Our project addresses these limitations by exploring a *diegetic approach*[1] to spatial audio accessibility. Rather than relying on non-diegetic HUD elements, the proposed system embeds visual cues directly into the 3D environment. By utilizing *real-time acoustic ray-casting* to approximate physical sound propagation, the developed Unity package maps sound information onto the surrounding geometry. This design allows players to visually perceive how sound travels through the virtual environment, preserving *spatial presence* while minimizing the user interface clutter.

2 Related Work

To contextualize the need for diegetic sound visualization, we categorize existing audio accessibility features by their representational abstraction: semantic, symbolic, and geometric.

2.1 Semantic and Text-Based Solutions

Early accessibility in 3D gaming relied primarily on linguistic substitutes. *Half-Life 2* (2004) is frequently cited for its comprehensive closed captions that substitute for environmental audio cues (see Figure 1). While captions are effective for narrative delivery, they do not convey spatial data quickly enough for real-time gameplay. As

*536425@mail.muni.cz
†jchmelik@mail.muni.cz
‡vit.rusnak@mail.muni.cz

[1] In video game UI design, diegetic elements are embedded directly into the 3D virtual environment rather than existing as external, screen-space overlays.

Coutinho et al. [2] demonstrated through semiotic inspection, text inherently struggles to rapidly convey directionality, distance, or the urgency of a threat. Consequently, relying solely on closed captions in fast-paced environments such as First-Person Shooters (FPS) leaves DHH players at a fundamental spatial disadvantage.

2.2 Symbolic Overlays and the Split-Attention Effect

To address the need for immediate spatial feedback, modern titles often utilize 2D Heads-Up Display (HUD) visualizers to convey spatial data. Many accessible games now convey essential information through redundant sensory modalities, combining visual screen changes with haptic controller vibrations to ensure events are noticed without audio. The radial indicator in *Fortnite* [1] is the most prominent example, mapping acoustic events onto a 2D ring (see Fig. 1, left). While functionally effective, this symbolic approach introduces a **split-attention effect**. Players must mentally translate 2D icons into 3D spatial coordinates, a process that can diminish **spatial presence** and increase task-related cognitive load [4].

2.3 Geometric and Diegetic Visualization

Recognizing the limitations of 2D overlays, recent research, particularly in fully immersive environments like Virtual Reality (VR), has explored anchoring sound data directly within the 3D environment. Systems such as SoundViz [5] utilize "On-Object Indicators" (see Fig. 1, right), which place text and icons directly on the virtual objects that generate the noise to visualize loudness, duration, and exact spatial location. While this effectively maps the origin of a sound diegetically, it relies on direct tagging of the source rather than simulating the physical behavior of the audio waves. Moreover, these often lack environmental context, failing to show how sound interacts with the scene.

2.4 Physical Sound Propagation and Geometric Acoustics

While on-object indicators successfully anchor sound to a 3D location, they often lack the environmental context of how that sound travels to the player. A more physically accurate approach involves simulating the actual *propagation of sound waves* through the virtual environment.

A leading example of this methodology is Vercidium Audio [9], a high-performance ray-traced acoustics engine designed to simulate real-time occlusion, diffraction, and reverberation. While the engine is currently in active development and not yet fully finished, it already demonstrates an advanced approach to spatial sound. Rather than treating audio as a simple point-to-point calculation, Vercidium utilizes *geometric acoustics*, casting thousands of audio rays into the environment to model complex behavior based on surface material properties (e.g., absorption coefficients). Vercidium has demonstrated that this data can be repurposed for accessibility by rendering shapes at **acoustic-geometry intersection points**, effectively "painting" the sound's path directly onto the world, allowing DHH players to perceive how sound navigates around corners and obstacles visually.

While Vercidium serves as the primary inspiration for our work, implementing a full-scale acoustic simulation engine introduces significant computational overhead and integration complexity. Our project aims to distill the core concept of ray-casted visual accessibility – *acoustic-geometry intersections* – into a lightweight, highly integrable visual framework. By focusing strictly on the *diegetic visual feedback* rather than full physical acoustic rendering, we provide a performance-conscious tool that developers can adopt as a "drop-in" solution without overhauling their existing audio architecture. By "painting" the sound's path across the environment, we bridge the gap between abstract symbolic cues and literal source tagging, providing a more intuitive representation of how sound navigates complex 3D geometry.

3 Project objectives

The goal of this research is to develop a framework that, in place of a traditional 2D HUD, visualizes acoustical events directly within the 3D environment. To achieve this, the system was designed around three core technical and accessibility requirements:

- *Spatial Fidelity:* To implement a physically-based simulation where visual cues coincide with the intersection of **acoustic wavefronts** and scene geometry. This ensures that the visual representation accurately reflects the sound's origin and its interaction with environmental obstacles.
- *Diegetic Integration:* To embed visual feedback directly into the game world to maintain *spatial presence*. By ensuring cues are part of the 3D scene rather than the interface layer, the system aims to reduce the *split-attention effect* and preserve player immersion.
- *Performance Viability:* To create a lightweight, modular Unity package that processes real-time *audio triggers* without significant computational overhead. The objective is to provide a "drop-in" solution that is accessible to independent developers without requiring a total overhaul of the existing audio architecture.

4 Design

To drive the diegetic visualizer, our system must simulate sound propagation dynamically within the 3D environment. We selected a ray-casting approach to model this

Figure 1: Examples of in-game sound visualizations. Left: Fortnite – circular orange HUD element around the center crosshair for visualizing sound; Middle: Half-Life 2 – sounds rendered as closed captions; Right: SoundViz – on-object sound indicators.

acoustic behavior. Geometric acoustic modeling, particularly ray tracing, has long been established as a standard technique for simulating spatial acoustics due to its effective balance of computational efficiency and perceptual accuracy [7]. While strict wave-based acoustic simulations offer high physical fidelity, their computational cost makes them prohibitive for real-time rendering. Conversely, ray-casting provides a performance-conscious framework capable of meeting the strict latency and interactivity requirements of modern video games [8]. This allows our system to provide immediate, spatially accurate visual feedback without severely impacting the game's overall frame rate.

4.1 System Architecture and Optimization

The primary technical challenge in simulating physical sound propagation lies in balancing computational overhead with visual accuracy. While a listener-centric approach (casting rays outward from the player) was explored, it failed to provide consistent environmental context. The proposed system employs a source-centric ray-casting architecture (see Figure 2). By emitting simulated rays directly from active sound sources, the system accurately traces the specular trajectories and interactions of acoustic wavefronts within the 3D environment.

To ensure consistent performance, the implementation utilizes a weighted-ray-budget scheduling. We employ custom shader optimizations for rendering visual indicators and take a source-focused approach to guarantee high sampling density for nearby acoustic events. This method also maintains a stable frame rate through asynchronous multi-threading.

4.2 The Ray-Casting Pipeline

The core simulation operates through a continuous, frame-by-frame pipeline that dynamically approximates sound wave propagation. The process is divided into three primary stages shown in Figure 3:

1. **Source Registry:** The system continuously polls the environment to identify all active audio sources. To prevent unnecessary calculations, only sources located within a predefined scan radius (R_{scan}) relative to the listener are registered for simulation.

2. **Ray Distribution:** To guarantee stable performance, the system enforces a strict, global maximum ray budget per iteration. This total budget is evenly distributed among the registered active sound sources. From each source, the allotted rays are cast outward in randomized spherical directions.

3. **Propagation, Collision, and Penetration:** As rays intersect with the 3D geometry of the environment, they register collision points. Upon impact, a probability check determines whether the ray bounces off the surface or passes through it. Crucially, a ray can reflect or penetrate, but not both simultaneously; this design choice limits the total number of rays in the scene, preventing the exponential growth of simulated paths often encountered in geometric acoustic modeling [7] and maintaining a strict computational budget. The probability of the ray reflecting is determined by a constant (ρ_{ref}) assigned to the specific collision layer the ray is colliding with. If reflected, an absorption factor (e_{loss}) is applied to simulate the absorption of acoustic energy at the impact site. Furthermore, to simulate natural sound decay in open

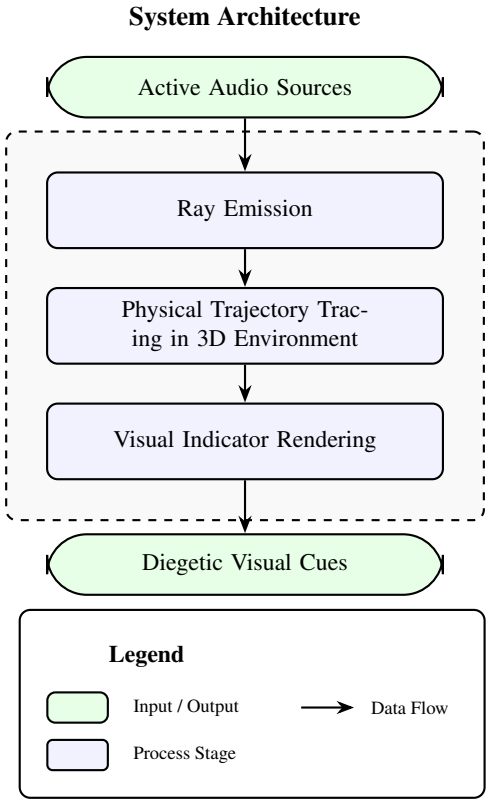

Figure 2: Data flow within the source-centric ray-casting architecture.

space, the ray loses energy continuously based on the distance traveled between bounces, calculated as:

$$E_{new} = E_{current} - (d_{air} \times \alpha_{air})$$

where d_{air} is the distance traveled through the air and α_{air} is the configurable air attenuation coefficient. A ray continues to propagate until its energy falls below a minimum threshold, it reaches a hard-capped bounce limit, or its cumulative travel length exceeds the maximum audible range of the audio source (D_{max}).

4.3 Sound Transmission and Acoustic Penetration

Basic ray-casting often assumes all geometry is fully occluding. To account for how sound actually transmits through objects based on material and thickness, the system uses a so-called "deferred backcasting" technique[2]. Because real-time interactive acoustic engines require strict CPU budgeting to maintain performance [8], the simulation relies on batched, asynchronous physics queries; a ray that passes through an object (as determined in the previous step) does not resolve immediately. Instead, its state is updated to *ResolvingPenetration*, and the transmission is calculated across a multi-step sequence:

1. **Forward Stepping:** Upon initial collision, the ray's origin jumps forward along its directional path by a predefined maximum expected wall thickness (W_{max}). This represents the maximum physical depth a sound wave can travel through a solid object before its acoustic energy is assumed to be entirely dissipated.

2. **Backcasting for Depth Calculation:** In the subsequent simulation frame, a secondary ray is cast backward from this advanced position toward the original impact point. If the backward ray returns a hit, the system records the distance it traveled through open space (d_{hit}). The exact thickness of the solid obstacle (d) is then approximated by subtracting this backward travel distance from the total maximum expected wall thickness:

$$d = W_{max} - d_{hit}$$

3. **Absorption and Culling:** If the backward ray fails to hit the geometry's exit face, it generally indicates that the advanced starting position is still physically inside the solid object (i.e., the object's thickness exceeds W_{max}). In this scenario, the sound is considered fully absorbed by the environment, and the ray is flagged as *Dead* to save computational resources.

4. **Material-Specific Attenuation:** If the backcast is successful, the system calculates the exact energy loss. To differentiate acoustic absorption across various materials (e.g., a thin wooden door versus a thick stone wall), the system allows developers to map specific acoustic properties—such as attenuation coefficients (α_{layer}) and reflection probabilities (ρ_{ref})—to distinguish environmental collision layers. This design ensures that the physical properties of the virtual environment determine acoustic energy loss, enabling easy integration with existing collision systems. The energy reduction is calculated as:

$$E_{new} = E_{current} - (d \times \alpha_{layer})$$

where d is the calculated thickness and α_{layer} is the energy loss per unit of distance for that specific material layer.

5. **Continuation and Epsilon Offset:** If the remaining energy (E_{new}) stays above a configurable minimum threshold (τ_{min}), the ray continues its forward path.

[2]Deferred backcasting is a multi-pass method used to calculate the thickness of hollow colliders. Because standard raycasts only detect front faces, the system schedules a delayed (deferred) secondary ray that fires backward from inside the object to hit its inner back face. The distance between the initial entry hit and this reverse exit hit yields the exact thickness needed for acoustic attenuation.

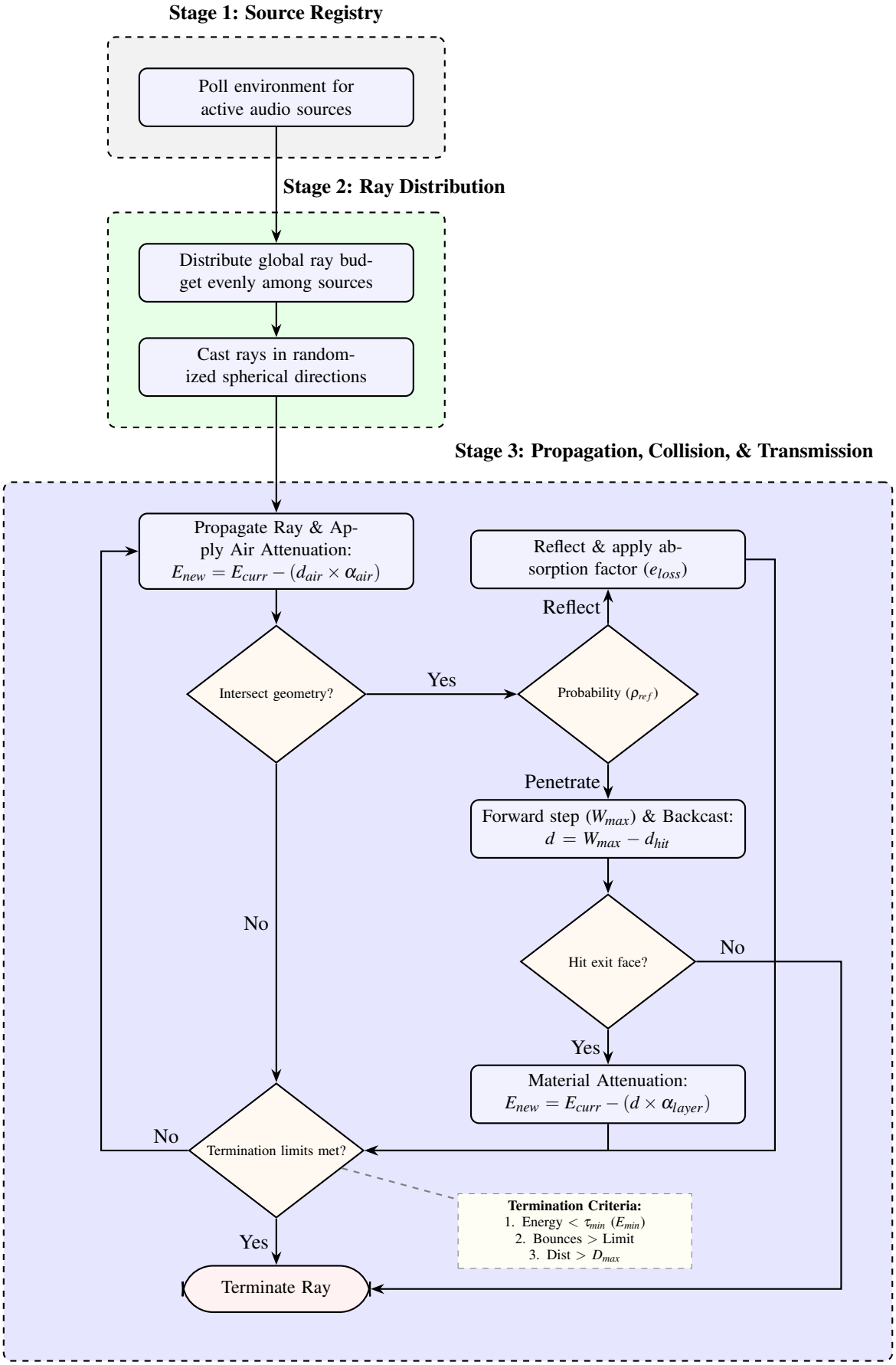

Figure 3: The ray-casting simulation pipeline.

This deferred backcasting approach effectively simulates physical sound dampening through walls while maintaining a strict computational budget, ensuring that visual indicators on the other side of an obstacle accurately reflect the muffled nature of the audio source. Notably, the backward ray does not need to strike the exact same object it originally penetrated; it simply needs to register a hit against any valid geometry to successfully approximate the spatial depth.

4.4 Visual Parameterization

The collision data generated by the ray-casting pipeline must be translated into intuitive visual cues for the player. Rather than drawing uniform indicators at every collision point, the system dynamically calculates the appearance of each visual cue based on the physical properties of its corresponding ray:

- **Rendering Threshold:** A visual indicator is only instantiated at a collision point if the ray's remaining energy exceeds the minimum threshold τ_{min}, ensuring that heavily muffled or distant sounds do not clutter the visual field.
- **Size and Scale:** The physical size of the indicator is proportional to the ray's energy. Direct, close-range sounds produce larger indicators, while heavily occluded sounds produce smaller, more diffuse cues.
- **Opacity and Lifespan:** To simulate the fading nature of sound, the opacity of each indicator decays over its lifetime. Additionally, the base intensity (I) scales inversely with the distance between the listener and the original sound source (d_{ls}). This is calculated as a linear falloff, clamped so that it drops to zero exactly as the listener reaches the sound source's maximum audible range (R_{max}):

$$I = \max\left(0, 1 - \frac{d_{ls}}{R_{max}}\right)$$

  This ensures that visual cues remain proportional to the auditory experience, naturally drawing the player's focus to immediate, proximate sounds while fading out distant noises.
- **Semantic Color Coding:** To differentiate between varying gameplay events (e.g., distinguishing footsteps from gunfire), the color of the indicator is inherited directly from a complementary tracking component attached to the original sound source defined by the user.

5 Results

The final outcome of this project is a fully functional Unity plugin that presents sounds as visual clues directly in the 3D scene, rather than a 2D HUD. As illustrated in Figures 4 and 5, the listener's perspective within the demonstration scenes is denoted by a camera object, while the active audio sources are represented by a speaker icon.

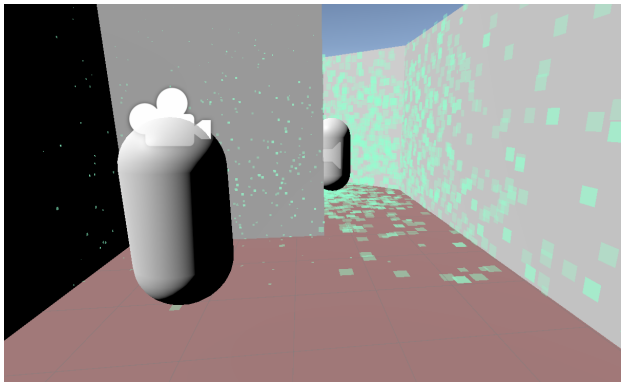

Figure 4: In-game demonstration of the ray-traced visualizer mapping sound propagation to surrounding surfaces.

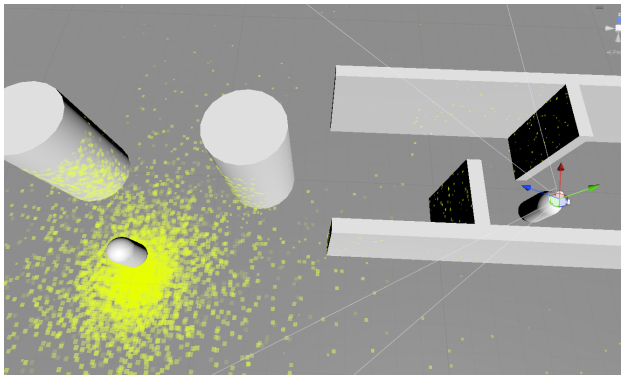

Figure 5: Alternative perspective demonstrating the distribution of the visual cues across varying environmental geometry.

5.1 Visual and Diegetic Integration

The prototype successfully maps sound directly onto the environment's geometry. During initial developer testing, the system demonstrated the visual tracking of sound as it bounced through doorways and around obstacles. Because the visual indicators inherit color coding from their respective sound sources and scale their opacity and size based on distance, the resulting cues carry important spatial information. This approach also allows players to perceive muffled audio through walls, calculated using the backcasting method.

5.2 Performance Viability

To evaluate the computational cost and scalability of the proposed ray-casting method, performance profiling was

conducted over a 2000-frame sample. The testing environment featured a baseline scene containing 15 dynamic sound sources, each emitting an audio signal every 250 ms. This baseline was compared against an identical scene running the RaytracedAudioVisualizer plugin under three distinct ray budget configurations: 100, 500, and 2500 rays per frame. Other configurations remained unchanged between tests.

As illustrated in Figure 6, the system demonstrated excellent scalability: a 100-ray budget introduced a 1.04 ms overhead, a 500-ray budget added 1.34 ms, and stress-testing at 2500 rays yielded an overhead of only 2.23 ms (2.59 ms total frame time). Additionally, preliminary testing during development suggested that the baseline budget of 100 rays per frame is already sufficient to produce clear and effective visualizations. This minimal, stable overhead confirms the visualizer is a viable, drop-in solution that comfortably operates within the 16.6 ms budget required for a 60 FPS rendering target.

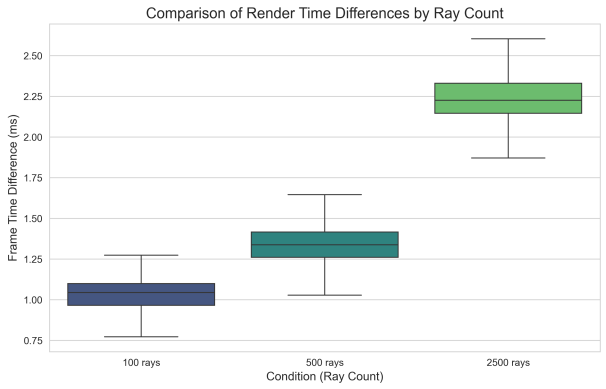

Figure 6: Performance comparison across different ray budgets.

5.3 Configuration

To allow developers to balance performance with visual fidelity, the *RaytracedAudioVisualizer* script exposes several configurable parameters organized into distinct categories. These settings control the ray-casting budget, the physical behavior of the simulation, and the visual output:

1. **General Settings:**
    - **Scan Radius:** (R_{scan}) Sets the maximum distance from the listener within which sound sources are actively tracked.
    - **Ray Count:** Determines the total budget of rays cast per frame to manage CPU load.
    - **Max Bounces:** Establishes a hard limit on how many times a single ray can reflect off geometry.
    - **Bounce Energy Loss:** (e_{loss}) An absorption factor applied to a ray's energy after each bounce to simulate absorption.
2. **Acoustics:**
    - **Max Expected Wall Thickness** (W_{max}) defines the maximum physical depth the system checks during deferred backcasting before assuming the sound is entirely absorbed.
    - **Min Energy Threshold** (τ_{min}) sets the minimum energy level required for an acoustic ray to continue propagating or spawning indicators; rays falling below this value are discarded.
    - **Air Attenuation Per Unit** (α_{air}) defines the natural loss of a ray's energy over distance as it travels through open space.
    - **Layer Configurations** allows developers to assign specific acoustic properties—such as customized attenuation rates (α_{layer}) and reflection/penetration probabilities (ρ_{ref}) to individual collision layers.

6 Limitations

While the current prototype successfully demonstrates the viability of diegetic, ray-casted audio visualization, it presents several limitations regarding aesthetic integration, technical compatibility, and physical acoustic accuracy.

6.1 Aesthetic Integration and Spatial Context

The system's current evaluation is limited to a controlled environment using basic primitive quads for visualization. While functional, this high-contrast representation can lead to *visual noise* or a lack of diegetic consistency in games with complex art styles. Furthermore, because ray-casting clarity and performance depend heavily on environmental geometry, the tool's effectiveness across diverse architectures remains untested.

6.2 Physical Simulation Accuracy

The reliance on *geometric acoustics* (ray-casting) inherently limits the physical fidelity of the simulation. Because geometric models approximate sound propagation as linear rays rather than spherical waves, they currently fail to simulate wave diffraction [7]. Consequently, visual cues do not naturally "bend" around corners or through narrow openings, which may lead to a mismatch between the visual cue's location and the perceived acoustic origin.

6.3 Technical and Pipeline Compatibility

A significant technical constraint is the plugin's tight coupling with Unity's native `AudioSource` components. Professional game development often uses external audio middleware, such as FMOD or Wwise, to handle complex logic. In its current state, the plugin lacks the API abstraction necessary to intercept and process audio data from these industry-standard third-party engines.

6.4 User Evaluation

At present, the system's effectiveness is justified through technical benchmarks rather than empirical human-subject data. No formal empirical data exists regarding how the target demographic interacts with these visualizations. It remains unverified whether the information density required for spatial accuracy induces cognitive overload or sensory fatigue during high-intensity gameplay. Formal testing is required to determine the optimal balance between visual detail and player performance.

7 Conclusion and Future Work

This paper presented a novel framework for spatial audio accessibility that shifts the paradigm from traditional non-diegetic HUD overlays to in-situ visual representations. By implementing a source-centric acoustic ray-casting architecture, the developed prototype successfully approximates physical sound propagation, mapping acoustic intensity and directionality directly onto the environmental geometry.

The resulting system demonstrates that transitioning spatial information from screen-space interfaces to diegetic 3D cues provides a viable path toward more inclusive game design. Unlike abstract icons, these world-anchored markers preserve the player's spatial presence while restoring critical gameplay feedback for Deaf and Hard-of-Hearing (DHH) users. By delivering a performance-conscious Unity package, this work offers developers a low-friction solution for integrating accurate accessibility features without compromising the visual integrity of the virtual world.

As this project transitions into a bachelor's thesis, subsequent research and development will focus on two primary trajectories: technical optimization and empirical validation. On the technical side, we will prioritize refining the ray propagation logic to address current physical approximations, improving the aesthetics of rendered cues, and expanding technical compatibility.

To empirically evaluate the usability of the proposed system, we plan to conduct a user study with both the hearing and the DHH users. These findings will be vital for determining whether diegetic visualization effectively conveys complex spatial information without inducing sensory overload, ultimately establishing a standardized design pattern for 3D audio accessibility.

References

[1] Accessibility Labs, LLC. Feature highlight: Fortnite's sound visualizer. Accessibility Labs: Case Studies, `https://accessibility-labs.com/feature-highlight-fortnites-sound-visualizer/`, 2025.

[2] Flávio Coutinho, Raquel O. Prates, and Luiz Chaimowicz. An analysis of information conveyed through audio in an fps game and its impact on deaf players experience. In *2011 Brazilian Symposium on Games and Digital Entertainment*, pages 53–62, 2011.

[3] Game Accessibility Guidelines. Ensure no essential information is conveyed by sounds alone. `https://gameaccessibilityguidelines.com/ensure-no-essential-information-is-conveyed-by-sounds-alone/`, 2012.

[4] Jasmine Granados, Dominic Canare, and Lisa Vangsness. Level-up! comparing accessibility features based on gameplay performance. *Journal of Accessibility and Design for All*, 14(2):1–15, 2024.

[5] Ziming Li, Shannon Connell, Wendy Dannels, and Roshan Peiris. Soundvizvr: Sound indicators for accessible sounds in virtual reality for deaf or hard-of-hearing users. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 2022.

[6] Sanzida Mojib Luna, Jiangnan Xu, Elise Baron, Garreth W. Tigwell, and Konstantinos Papangelis. Motivation and re-engagement in mixed reality: How deaf and hard of hearing users experience a mixed reality exergame. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 2025.

[7] Lauri Savioja and U Peter Svensson. Overview of geometrical room acoustic modeling techniques. *The Journal of the Acoustical Society of America*, 138(2):708–730, 2015.

[8] Carl Schissler and Dinesh Manocha. Gsound: Interactive sound propagation for games. In *Audio Engineering Society Conference: 41st International Conference: Audio for Games*. Audio Engineering Society, 2011.

[9] Vercidium. Vercidium audio: High-performance, low-latency audio for web applications. `https://vercidium.com/audio`, 2026.

[10] World Health Organization. Deafness and hearing loss. WHO Fact Sheet, `https://www.who.int/news-room/fact-sheets/detail/deafness-and-hearing-loss`, 2025.