Audio Engineering Society

Conference Paper 6

Presented at the AES International Conference on
Headphone Technology
2025 August 27–29, Espoo, Finland

This paper was peer-reviewed as a complete manuscript for presentation at this conference. This paper is available in the AES E-Library (http://www.aes.org/e-lib), all rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

RoomAcoustiC++: An open-source room acoustic model for real-time audio simulations

Joshua Mannall[1], Lauri Savioja[2], Annika Neidhardt[1], Russell Mason[1], and Enzo De Sena[1]

[1]*Institute of Sound Recording, University of Surrey*
[2]*Department of Computer Science, Aalto University*

Correspondence should be addressed to Joshua Mannall (j.mannall@surrey.ac.uk)

ABSTRACT

An open-source C++ library named RoomAcoustiC++: Real-time Acoustics Library for real-time room acoustic modelling is introduced that implements a hybrid geometric acoustic and feedback delay network model. The geometric acoustic component uses the image edge model to locate geometry-dependent early specular reflections and edge diffraction. The feedback delay network models late reverberation for a target reverberation time. The model is capable of a dynamic simulation including moving geometry, sources and listener and changing wall absorption, with binaural spatialisation over headphones and customisable head-related transfer functions using the 3D Tune-In toolbox. A comparison with existing closed-source and open-source projects is presented. This found that many state-of-the-art room acoustic models for real-time applications are closed-source, limiting reproducibility. RoomAcoustiC++ offers an improved room acoustic model compared to existing open-source projects. The library was validated against physical measurements from the Benchmark for Room Acoustic Simulations (BRAS) database. An analysis of the real-time performance shows that the software is capable of binaural rendering for scenes with occluding geometries and multiple sources.

1 Introduction

Many tools exist for simulating acoustic scenes offline, such as CATT-Acoustic [1], Odeon [2], Pyroomacoustics [3], RAVEN [4], Treble [5] and COMSOL [6]. These tools are well suited to the building and automotive acoustics industries but have limited support for real-time six degrees of freedom (6DoF) audio rendering where the acoustic scene is updated as the source, receiver or geometry of the physical scene changes. The demand for real-time acoustic modelling has been driven by the desire for immersive gaming and extended reality (XR) experiences. For example, a binaurally rendered reverberation model can improve a participant's sense of immersion within virtual reality (VR) [7].

Immersive audio for XR applications requires an acoustic model capable of real-time processing with 6DoF. The implementation of this poses many challenges, such as memory and thread management, ensuring low-latency performance and ren-

dering dynamic scenes without audible clicks. Several commercial products are available, such as the Meta XR Audio SDK [8], AudioKinetic: Reflect [9], and Project Acoustics [10]. These tools are designed for the game industry, where performance is critical. Therefore, the fidelity of the acoustic model is often limited. Being commercial products, they are not distributed under an open-source licence, which prevents researchers from customising software to their specific use case and lacks the details required for reproducibility.

This paper introduces an open-source C++ library named RoomAcoustiC++: Real-time Acoustics Library[1] designed to provide researchers with a well documented, open-source room acoustic model for real-time applications. The remainder of this paper is structured as follows. Section 2 provides a high-level overview of room acoustic modelling. This focusses on geometrical acoustics (GA) modelling which is the most common choice for real-time room acoustics. Section 3 reviews various existing frameworks and compares them against RoomAcoustiC++ (RAC). Section 4 describes the RAC framework for each stage of the acoustic model and the audio pipeline, along with specific implementation details. Section 5 validates the model against the Benchmark for Room Acoustical Simulation (BRAS) database.

2 Background

An overview of current room acoustic modelling techniques is given in [11]. Two important classes of models are wave-based and ray-based modelling, whereby the former attempts to solve the wave equation for given source, receiver and geometry conditions. This accurately captures effects such as sound propagation, scattering and diffraction; however, it is computationally expensive, ill-suited to dynamic geometry and scales poorly with scene volume. Because of this, most real-time acoustic simulations are ray-based, using GA to approximate sound propagation from a source as geometric rays. GA approaches often divide the modelling process into discrete tasks, such as locating early reflections and determining late reverberation parameters. Models in this class include image-source based methods [12], ray tracing [13] and beam tracing [14].

2.1 Geometrical acoustics

The image-source (IS) model is described in [15, 12]. The main principle is to locate ISs by recursively reflecting sources in walls. The image-edge (IE) model [16], an extension to the IS model, was proposed by Erraji et al. to incorporate diffraction paths. Combined reflection and diffraction paths are found by reflecting edges along with sources. These processes guarantee that all paths up to a specified order are found [16]. However, the processing time increases significantly with increasing orders of reflection and diffraction. This makes these models challenging for late reverberation modelling. Computation of the IS model can be sped up by beam-tracing [17, 14] and has been applied to diffraction in 2D scenes [18].

Every IS describes a position in 3D space and a frequency dependence based on reflections and diffractions encountered on its path through the scene. The contribution of a single IS to the room impulse response (RIR) can be expressed as

$$y(t) = \frac{(h * x)\left[t - \frac{r}{c}\right]}{r}, \tag{1}$$

where $h[\cdot]$ is a frequency-dependent filter, $x[\cdot]$ is the anechoic input signal, c is the speed of sound, r is the distance from the receiver and $*$ is the convolution operator. The IS can be spatialised using panning methods, such as vector based amplitude panning (VBAP) [19], or by applying an head-related transfer function (HRTF) to find the binaural response [20].

Ray tracing [21, 22] uses principles similar to the traditional IS model. In this case, reflection paths are found by emitting multiple rays from a source that then reflect within the scene. Both diffuse and specular reflections can be modelled [22]. Valid paths are located when the rays intersect with the receiver volume and can be used to generate an echogram or locate ISs. This approach does not guarantee finding all early reflections. However, at high reflection orders, it is significantly more efficient than the IS model. Approaches to incorporate

[1] https://github.com/jmannall/RoomAcoustiCpp

diffraction within a ray tracing model include considering rays that intersect a face near the edge [23] or pre-computing visibility between edges [24, 13]. Neither of these implementations consider combined reflection and diffraction paths.

2.2 Late reverberation

Unlike early reflections and diffraction, individual late reflections are less critical to the perception of an acoustic environment [25]. Therefore, late reverberation models are designed to model acoustic attributes such as reverberation time and modal density. While several late reverberation models exist [26], feedback delay networks (FDNs) are amongst the most used. An FDN is a recursive system that uses a feedback matrix to distribute energy across multi-channel delay lines. A common challenge is tuning the FDN parameters to achieve the desired acoustic parameters. Bai et al. [27] propose using the acoustic radiance transfer method (RTM) to link the feedback delay lines within an FDN to the energy exchange between surface patches. FDNs have been applied to coupled volumes [28, 29] by combining networks to model the transfer of energy between rooms.

3 Related work

Table 1 compares nine existing real-time libraries with RAC and summarises their key features.

3.1 Commercial projects

Reflect [9] and RoomVerb [30] by Audiokinetic provide one of the most comprehensive commercial solutions. Early reflections and diffraction are modelled using an IE model [16]. However, diffraction effects are not physically accurate and are modelled using a simple gain and low-pass filter (LPF) cut-off frequency based on the bending angle. Late reverberation is not modelled based on the scene geometry and must be manually tuned per room with controls such as decay time, room shape, reverb level and density. The acoustic models are well documented at a high level and many controls, such as distance attenuation and reflection order, are exposed to the user. This makes it highly customisable from a games design perspective. It requires knowledge of the Wwise middleware software and a paid licence if a standalone application is needed. An open-source licence is not available.

The Meta XR Audio SDK [8] utilises a highly optimised ray tracing engine for geometry-dependent reverberation and diffraction. Ray tracing is performed offline and creates an acoustic map of points that are interpolated between at runtime. Specific implementation details are not available.

Project Acoustics [10] uses the wave-based adaptive rectangular decomposition (ARD) model [31] to simulate the acoustics of a scene. It requires a computationally expensive pre-computation with large GPU or cloud compute resources. The baked directional impulse responses are interpolated at runtime, and therefore the application to dynamic geometry is limited. Individual bakes can be assigned to differentiate open or closed doors. Research from Project Acoustics has been published, however the tool is restricted to internal Microsoft Game studios and is not released open source.

3.2 Open source projects

The 3DTI Toolbox [32] was developed as an open-source software with multiple platform support, dynamic binaural rendering and customisable HRTFs. The tool focusses on binaural spatialisation and hearing loss and hearing aid simulation [33]. The room acoustic capabilities are limited to precomputed binaural room impulse responses (BRIRs).

Planeverb [34] was developed to demonstrate how a wave-based model could be processed in real-time with moving sources, receivers and geometry. This was achieved by running a 2D simulation on a horizontal slice of the geometry at the listener's head height. This provides convincing results in many situations, but the authors note that this relies on all acoustically significant geometry intersecting the 2D plane. They also stated that the 2D to 3D correction requires further research.

EVERTims [35] was developed by researchers at LIMSI/CNRS, the TKK/Department of Media Technology and the IRCAM institute. It uses beam tracing to locate early reflections [36] and Sabine's formulae to predict the **RT60!** (**RT60!**) for late reverberation. This is implemented using an FDN.

The acoustic engine is built within the JUCE framework, and a user interface is available as an add-on for Blender. It is open source, but the Blender interface is not designed to integrate with exiting VR and augmented reality (AR) technologies. It does not include diffraction modelling.

TASCAR [37] is an open-source tool developed for applications in hearing aid research and audiology. It uses the IS model to locate early reflections and Sabine's formulae to predict the **RT60!** for late reverberation. This is implemented using an FDN. It does not include diffraction modelling.

3.3 Academic use

RAVEN [4] is a GA based model with a MATLAB and SketchUp interface. It can be used as a real-time model in SketchUp or paired with the Virtual Acoustics framework [38]and controlled using Unreal Engine, Unity, MATLAB or Python. It is available to use as is for research purposes and does not provide open-source access. There have been many versions of RAVEN (including diffraction models and coupled-room models), which are not featured in the current version.

liveRAZR [39] is a hybrid GA-FDN model developed at the University of Oldenburg. It uses the IE model to locate early reflections and diffraction path and controls reverberation time using an FDN. The software is available upon request from the authors. It is currently not available with an open-source licence.

4 Technical features

The following section outlines the technical details and implementations of the proposed RAC library. It provides a high-level overview of the software capabilities and structure. Then it describes the implemented acoustic models, audio processing design choices and software integration.

4.1 Overview

The acoustic model is based on the work by Wendt et al. [40], whereby an IS model feeds the input of an FDN. This couples geometry-dependent early reflections with an efficient late reverberation model. The model was evaluated by Wendt et al. through objective and perceptual measures for roughly shoebox-shaped rooms. In RAC, the IS model is replaced by the IE model [16].

The code base is divided into three independent threads: update, audio, and background. The update thread handles the creation and deletion of sources and walls as well as updates to the positions and orientations of the scene geometry, sources and receiver. The background thread reads the data from the update thread, runs the IE model to locate visible ISs and determines the absorption of the nearest wall in the direct of each FDN reflection filter. Then it writes to data accessed by the audio thread. The refresh rate of this thread depends on the time taken to run the models which varies based on the current IE configuration and the number of walls and sources. The IE model is summarised in Section 4.2. The audio thread receives a mono audio buffer and applies digital signal processing before returning a stereo buffer. Binaural rendering in RAC is handled by the 3D Tune-In (3DTI) Toolbox [32] which offers three settings for spatialisation: *quality* (HRTF), *performance* (ILD and ITD) and *none* (mono).

4.2 Image edge model

The background thread runs an IE model [16] using the most recent room geometry, source positions and listener position. The IE model settings can be modified at runtime and allows the direct, reflection and diffraction components to be toggled independently. Diffraction paths can be included either only when the receiver is in the shadow zone (e.g no direct sound) or in all zones around the edge. The IE model is used to update the list of visible ISs accessed by the audio thread.

The geometry of the scene is stored as a collection of walls, where a wall describes a single triangle (three vertices) with a given absorption property. Groups of coplanar walls are assigned to a plane as described in [41]. This allows collections of walls to be discounted if a line segment (e.g. the line between an IS and the receiver) does not cross the correct plane and speeds up validity and visibility checks. There is no requirement for watertight geometry. Ray-triangle intersections are found using the methodology described in [42]. Edges are created by locating shared vertices between triangles.

When diffraction paths are located, the apex point (the shortest path between the source and receiver via the edge) is used to test for visibility, and therefore visibility of the entire edge is not considered. Diffraction is currently only supported up to 1st-order diffraction, but a diffraction path can include any number of reflections.

4.3 Audio processing

The audio processing is built on the 3DTI toolbox, which handles spatialisation based on the relative positions of the receiver, source and any ISs. By placing ISs in appropriate positions, the additional processing required by RAC is limited to frequency dependent filtering and late reverberation processing. This process is outlined in Figure 1.

Each audio frame iterates over an array of sources and visible ISs. These can be processed in parallel using multiple threads. For each IS, relevant diffraction, reflection and air absorption filtering is applied before spatialisation. Numerous diffraction models have been implemented including Biot-Tolstoy-Medwin (BTM) [43], uniform theory of diffraction (UTD), implemented using 4th order Linkwitz-Riley crossover filters [24, 13], NN-IIR [44] and Universal Diffraction Filter Approximation (UDFA) [45]. The BTM model is processed using a finite impulse response (FIR) filter, while the rest make use of efficient infinite impulse response (IIR) filter structures. Air absorption filters are implemented as 1st-order IIR filters as described in [46]. All other frequency dependent filtering such as wall absorption and source directivity is implemented as multi-band graphic equalisers as described in [47]. The equaliser is built using a 2nd-order peaking IIR filter per filter band and an additional low-shelf and high-shelf filter to control the DC and Nyquist gain. This allows for custom absorption bands defined by the user at initialisation and a low-complexity filter for real-time processing. Source directivity can be set as common polar patterns (frequency-independent) or defined using spherical harmonics (per frequency band). Genelec 8020c and QSC K8 directivities have been implemented in this way using measurements from the BRAS database.

The receiver, source and geometry are able to move in real-time. Therefore, the audio processing must be able to interpolate between states without audible artefacts. Much of this is tied into the HRTF processing and handled by the 3DTI toolbox, including distance-based delays and attenuation. Within RAC, the interpolation of DSP parameters (for example gains, filter parameters and impulse responses) is carried out on a sample-by-sample basis using a linear interpolation. During experiments [48] run using this framework, no audible clicks were reported.

4.4 Late reverberation

Late reverberation is modelled using an FDN as proposed by Wendt et al. [40] whereby the inputs to the FDN are the highest order ISs and the reverberation is spatialised by distributing the output channels around the listener [25]. This distribution is determined by the number of FDN channels. In RAC, an axis-aligned platonic solid (or combination thereof) with an appropriate number of vertices is selected from a list. The position of each vertex creates a reverb source and is used by the 3DTI toolbox to spatialise each of the FDN channels. The FDN structure is shown in Figure 1. Each channel includes a delay and absorption filter which allows the reverberation time to be controlled. The output of each channel is linked to a reverb source and the reflection filters are set to the absorption of the nearest wall in the direct of the corresponding source. At runtime, the reverb sources are centred around the listener.

Similarly to [40], the delay lines are set based on distance. For a given distance (D), the delay line length (d) in samples is given as

$$d = \frac{f_s}{c}\left(D + \frac{U(-D_{mean}, D_{mean})}{10}\right), \qquad (2)$$

where c is the speed of sound, f_s is the sample rate, D_{mean} is the average of all the distances across the delay lines and U is a uniform distribution. This adds a small randomisation in the case of identical length delay lines. Following this, the delay line lengths are adjusted to be mutually prime. For shoebox-like rooms, reverberation time can be calculated using either Sabine's or Eyring's formula using the total wall absorption. Alternatively, it can be set manually, allowing for dynamic changes

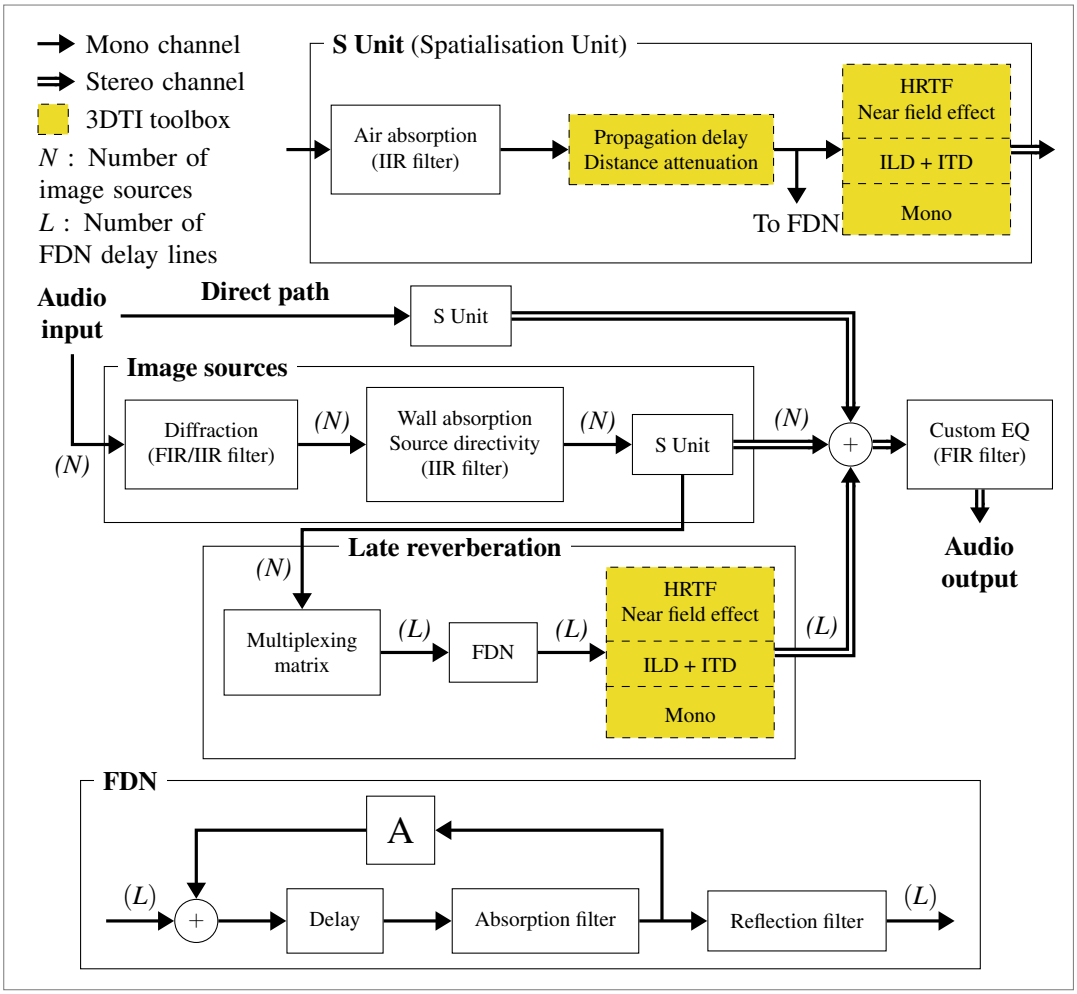


Fig. 1: An outline of the audio processing stage. When considering multiple sources the late reverberation section is only processed once after calculating ISs for each source.

at runtime. Room geometries with coupling behaviour or particularly complex geometry or absorption characteristics, where simple modelling using Sabine or Eyring is insufficient, would require a more advanced FDN design.

4.5 Integration

RAC is available through a public GitHub repository[2] as an open-source C++ library along with a Visual Studio project to build either a dynamic link library (DLL) for the Windows platform or a shared object (SO) for Linux/Android/MacOS platforms (but the latter has not been tested for). These libraries expose functions that control high-level functionality such as initialising RAC, updating source, receiver and geometry positions and modifying the IE model configuration.

RAC can be used either by (a) linking it as a library to a user's own custom code or (b) directly for rendering via Unity. A Unity template project[3] is provided that enables a simple interface with the libraries through modular C# scripts for sources,

[2] https://github.com/jmannall/RoomAcoustiCpp

[3] https://github.com/jmannall/RoomAcoustiCpp-Unity

listeners and geometry. It runs locally on the CPU. Hardware capabilities determine the achievable binaural rendering quality and the maximum reflection order. While it can run in real-time on hardware such as the Meta Quest Pro, there remains significant room for optimisation.

5 Performance and validation

This section presents an analysis of the computational performance of RAC using profiler tools, and a validation comparing its output with physical measurements from the BRAS database [49].

5.1 Computational performance

For real-time applications, RAC must be able to process each audio frame within the allocated time and update the acoustic scene with a low level of latency. The maximum processing time for typical frame sizes of 256 to 1024 samples and a sample rate of 48 kHz is 6 to 23 ms respectively. If the processing time exceeds this, a buffer underrun will occur and may cause audible artifacts. Larger frame sizes are advantageous as it allows the CPU to take greater advantage of optimisations for batch operations and reduces some of the overhead from per frame operations; however, increases throughput audio latency. The latency of the acoustic model is affected by the processing time of the background thread. As the complexity of the IE model and the number of sources increase, the processing time increases. This leads to increased latency between changes in the scene and updates to the acoustic model. In AR and VR, low-latency head rotations are critical [50]. The HRTF filters are selected in the audio thread; therefore, by updating the listener position and orientation of the audio listener, this latency can be decoupled from the acoustic model latency.

5.2 Profiling

All the tests in this section ran on a standard desktop PC with 32GB RAM, an AMD Ryzen 7 7700 CPU running Windows 11, 64-bit. The background thread was run single threaded while the audio processing was run multithreaded with 8 concurrent threads processing the early reflections and late reverb spatialisation. The code was profiled over a

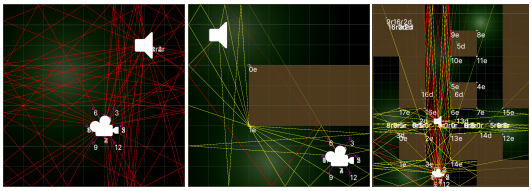


Fig. 2: Top-down view of the scenes used for profiling with example reflection (red) and diffraction (yellow) paths. From left to right: Shoebox, U-Room and Room 42.

10 s interval using a custom profiler (part of RAC) while running from the Unity interface. Unity was the only process running on the computer along with the operating system processes. The IE model included early reflections, shadowed diffraction and specular diffraction (up to 1st-order). All simulations were run using a sample rate of 48 kHz, a 12 channel FDN, six octave bands for frequency dependent filtering, BTM diffraction model, an HRTF length of 384 samples, five degree HRTF sampling and an interpolation factor of two.

Three scenes [39], shown in Figure 2, were chosen for evaluation. A shoebox room with 12 triangles and no edges, a U-shaped room with 44 triangles and 2 edges and a complex environment with 192 triangles and 18 edges termed Room 42. All evaluations were conducted with a moving listener (on a programmed path) to best represent a real use case. Simulations with a stationary listener ran approximately 4% faster as they avoided interpolation of delay lines and filter parameters.

Figure 3 shows the audio processing time for a single source in the shoebox room. Some outliers can be seen that can be attributed to the measurement setup. Like all general-purpose operating systems, Windows dictates that CPU time is shared between multiple processes. This can lead to the RAC process being interrupted by other system processes. Therefore, processing time here is defined as the time taken for RAC to process a single frame plus the time taken for any CPU interrupts. It can be seen that the *quality* (HRTF) setting requires more resources to run compared to the *performance* (ILD + ITD) setting, and that the processing time increases with the maximum reflection order due to the increasing number of ISs.

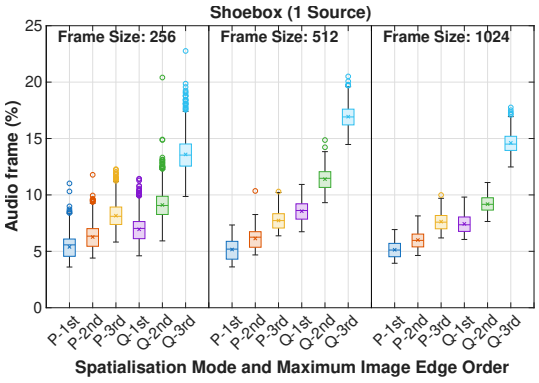


Fig. 3: Processing time on the audio thread for multiple frame sizes for a shoebox room with one source. This is expressed as a percentage of the audio frame time length (a value above 100% would indicate a buffer underrun). Along the x-axis, P and Q represent the *performance* or *quality* mode respectively followed by the maximum IE order. The mean is marked by a cross.

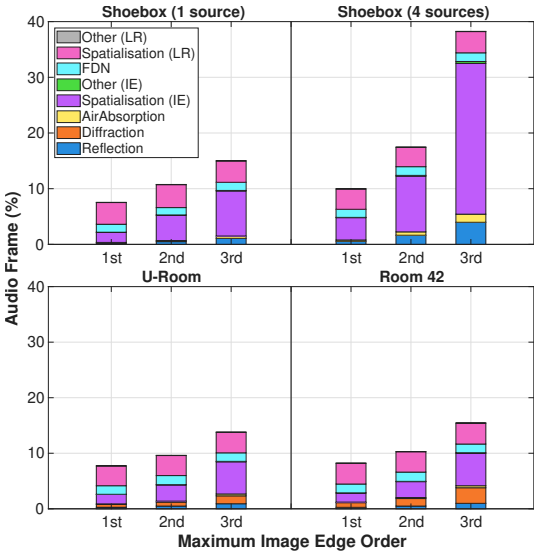


Fig. 4: Mean processing time on the audio thread split by task. Run using a frame size of 1024 and with binaural processing (*quality* mode). IE and LR denote image edge and late reverberation respectively.

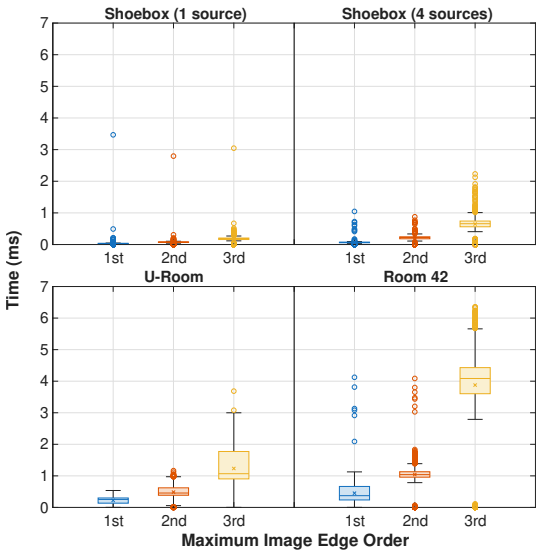


Fig. 5: Processing time on the background thread for multiple scenes and maximum IE orders. The mean is marked by a cross.

Figure 4 shows a breakdown of the different processing tasks. It is clear that binaural processing accounts for a large proportion of the processing time. Increasing the number of sources has a significant impact on performance due to the (linearly) increasing number of ISs. This is most significant at high reflection orders, when the number of ISs per source is highest. The time taken for late reverberation remains constant as this is processed once per frame regardless of the number of sources, hence why the increase of complexity from one to four sources is less than four times. Increasing the number of ISs also improves the multithreading efficiency gains. The U-Room and Room 42 involve additional computational complexity due to diffraction modelling, but due to fewer visible ISs, they yield a similar processing time compared to the Shoebox room. More efficient models than BTM (used here) have been implemented in RAC.

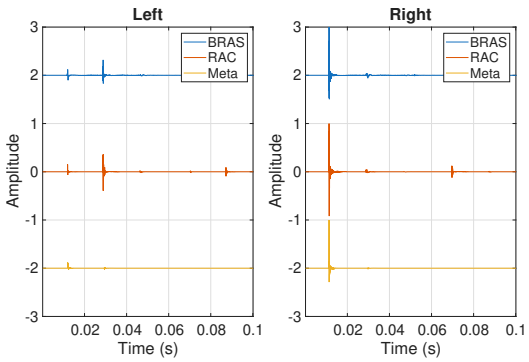
Figure 5 shows the total processing time for each iteration of the background thread. Including diffraction path finding, increasing the number of polygons, or increasing the maximum IE order have the biggest impact on the processing time. Room 42 shows a significant increase in the processing time for the IE model at 3rd-order which highlights

the branching nature of the IE model. Overall, the processing time remained below 7 ms, which is sufficient for interactive update rates. Due to being single-threaded, the processing time increases linearly with the number of sources. Therefore, in order to run the most complex scenes with multiple sources, the maximum IE order may have to be limited.

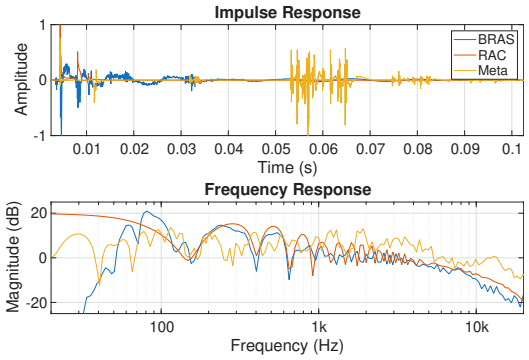
5.3 Accuracy

RAC was validated against the BRAS database (reference scenes 1-7 and complex room 2). While all these scenes were used, Figure 6 shows three examples from the data set that were selected to demonstrate the IE model in critical scenarios and the accuracy of the FDN energy decay in a reverberant room. The reference scenes (RS) were simulated using a 6th-order IE model with early reflections and specular and shadowed diffraction modelling. The complex room (CR) scene was modelled using a 3rd-order IE model and with specular diffraction limited to 1st-order paths. The Genelec 8020c loudspeaker directivity was used. The Meta XR Audio SDK (MetaSDK) [8] is included as a commercial baseline. The default settings were used with the acoustic map set to a maximum spacing of 2. For the RIR examples, the HRTF intensity was set to zero. It is not possible to fully remove the ILD and ITD modelling; therefore, the left channel was selected. Custom directivity patterns are also not possible.

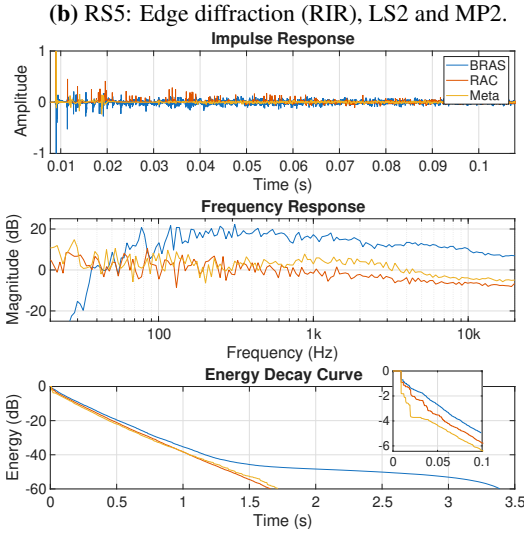
Figure 6a shows that RAC can accurately model a high number of early reflections. When listening to this BRIR, a clear flutter echo can be heard. The MetaSDK only located a 1st-order reflection. The frequency response of BRAS in Figure 6b shows notches and a high frequency roll-off that are also captured in the RAC simulation. The MetaSDK locates the early reflection paths although the notches are not clearly present in the frequency response. This may be due to the late reverberation-like tail (although late reverberation was toggled off for this simulation). In the complex room example, shown in Figure 6c, the RAC, BRAS and MetaSDK energy decay curves match closely. The MetaSDK shows a steeper early decay, although this may be due to only selecting the left channel.



(a) RS3: Parallel reflector (BRIR), LS1 and MP2.



(b) RS5: Edge diffraction (RIR), LS2 and MP2.



(c) CR2: Shoebox like room (RIR), LS1 (positive Y) and MP4.

Fig. 6: Comparison of RAC against scenes from the BRAS database. Frequency responses were smoothed using 16th octave bands.

Across all the scenes RAC shows close alignment with the BRAS database. Future perceptual experiments are planned to evaluate RAC in an AR environment. Previous studies using RAC [48] have shown that the model is capable of highly plausible acoustic simulations.

6 Conclusions

This paper has presented a 6DoF acoustic model suitable for real-time applications. It combines the IE model [16] with the late reverberation model proposed in [40]. It is provided open source, supports any room geometry and incorporates a variety of diffraction models.

It is primarily designed as a research tool and is released open source in the hope that it will enable and encourage further research into perception of room acoustics in XR. The modular Unity interface provides straightforward integration with current AR and VR technologies. This could benefit non-specialist researchers in the fields of psychology, medicine or engineering etc [51] where immersive environments are also important. This interface is currently being used to perform perceptual experiments at the University of Surrey in VR [48] and XR.

There is space for further feature development of RAC such as material specific scattering as described in [29] and higher-order diffraction modelling. A significant future aim is the development of a fully geometry-dependent late reverberation model. This could introduce double-slope decay modelling and expand the use case from single-room to multi-room environments. While RAC is suitable for real-time applications, there is still substantial room for performance optimisation. This would be most beneficial for deployment to AR and VR headsets.

7 Acknowledgement

This work was funded in part by a scholarship of the University of Surrey, the Engineering and Physical Sciences Research Council (EPSRC) under the 'SCalable Room Acoustics Modelling (SCReAM)' grant EP/V002554/1 and the 'Challenges in Immersive Audio Technology (CIAT)' grant EP/X032914/1.

References

[1] CATT, "CATT-Acoustics," 2024, https://www.catt.se.

[2] ODEON, "ODEON Room Acoustics Software," 2024, https://odeon.dk/.

[3] Scheibler, R., Bezzam, E., and Dokmanic, I., "Pyroomacoustics: A Python Package for Audio Room Simulation and Array Processing Algorithms," pp. 351–355, 2018.

[4] Schröder, D. and Vorländer, M., "RAVEN: A Real-Time Framework for the Auralization of Interactive Virtual Environments," in *Forum Acusticum*, pp. 1541–1546, 2011.

[5] Treble, "Treble Acoustic Simulation Platform," 2024, https://www.treble.tech/acoustics-suite.

[6] COMSOL, "COMSOL Multiphysics® Software," 2024, https://www.comsol.com/comsol-multiphysics.

[7] Potter, T., Cvetković, Z., and Sena, E. D., "On the Relative Importance of Visual and Spatial Audio Rendering on VR Immersion," *Front. Signal Process.*, 2022.

[8] Meta, "Meta XR Audio SDK (UPM)," 2023, https://developer.oculus.com/downloads/package/meta-xr-audio-sdk/.

[9] Audiokinetic, "Reflect," 2024, https://www.audiokinetic.com/en/products/plugins/reflect/.

[10] Microsoft, "Project Triton - Immersive sound propagation," 2023, https://www.microsoft.com/en-us/research/project/project-triton/.

[11] Savioja, L. and Svensson, U. P., "Overview of geometrical room acoustic modeling techniques," *J. Acoust. Soc. Amer.*, pp. 708–730, 2015.

[12] Borish, J., "Extension of the image model to arbitrary polyhedra," *J. Acoust. Soc. Amer.*, pp. 1827–1836, 1984.

[13] Schissler, C., Mückl, G., and Calamia, P. T., "Fast diffraction pathfinding for dynamic sound propagation," *ACM Trans. Graphic.*, pp. 1–13, 2021.

[14] Laine, S., Siltanen, S., Lokki, T., and Savioja, L., "Accelerated beam tracing algorithm," *Appl. Acoustics*, pp. 172–181, 2009.

[15] Allen, J. B. and Berkley, D. A., "Image method for efficiently simulating small-room acoustics," *J. Acoust. Soc. Amer.*, pp. 943–950, 1979.

[16] Erraji, A., Stienen, J., and Vorländer, M., "The image edge model," *Acta Acustica*, p. 17, 2021.

[17] Funkhouser, T., Tsingos, N., Carlbom, I., Elko, G., Sondhi, M., West, J. E., Pingali, G., Min, P., and Ngan, A., "A beam tracing method for interactive architectural acoustics," *J. Acoust. Soc. Amer.*, pp. 739–756, 2004.

[18] Antonacci, F., Sarti, A., and Tubaro, S., "Two-Dimensional Beam Tracing from Visibility Diagrams for Real-Time Acoustic Rendering," *EURASIP J. Advances in Signal Processing*, 2010.

[19] Pulkki, V., "Virtual Sound Source Positioning Using Vector Base Amplitude Panning," *J. Audio Eng. Soc.*, pp. 456–466, 1997.

[20] Blauert, J., *Spatial Hearing : The Psychophysics of Human Sound Localization*, MIT Press, revised edition edition, 1997.

[21] Krokstad, A., Strom, S., and Sørsdal, S., "Calculating the acoustical room response by the use of a ray tracing technique," *Journal of Sound and Vibration*, pp. 118–125, 1968.

[22] Vorlander, M., "Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm," *J. Acoust. Soc. Amer.*, pp. 172–178, 1989.

[23] Taylor, M., Chandak, A., Mo, Q., Lauterbach, C., Schissler, C., and Manocha, D., "Guided multiview ray tracing for fast auralization," *IEEE Trans. Visualization and Computer Graphics*, pp. 1797–1810, 2012.

[24] Schissler, C., Mehra, R., and Manocha, D., "High-order diffraction and diffuse reflections for interactive sound propagation in large environments," *ACM Trans. Graphic.*, pp. 1–12, 2014.

[25] Kirsch, C., Poppitz, J., Wendt, T., van de Par, S., and Ewert, S. D., "Spatial Resolution of Late Reverberation in Virtual Acoustic Environments," *Trends in Hearing*, 2021.

[26] Välimäki, V., Parker, J. D., Savioja, L., Smith, J. O., and Abel, J. S., "Fifty years of artificial reverberation," *IEEE Trans. Audio, Speech, Language Proc.*, pp. 1421–1448, 2012.

[27] Bai, H., Richard, G., and Daudet, L., "Late Reverberation Synthesis: From Radiance Transfer to Feedback Delay Networks," *IEEE Trans. Audio, Speech, Language Proc.*, pp. 2260–2271, 2015.

[28] Das, O., Schlecht, S. J., and Sena, E. D., "Grouped Feedback Delay Networks With Frequency-Dependent Coupling," *IEEE Trans. Audio, Speech, Language Proc.*, pp. 2004–2015, 2023.

[29] Kirsch, C., Wendt, T., Par, S. V. D., Hu, H., and Ewert, S. D., "Computationally-Efficient Simulation of Late Reverberation for Inhomogeneous Boundary Conditions and Coupled Rooms," *J. Audio Eng. Soc.*, pp. 186–201, 2023.

[30] Audiokinetic, "RoomVerb," 2024, `https://www.audiokinetic.com/en/library/edge/?source=Help\&id=wwise_roomverb_effect_plug_in`.

[31] Raghuvanshi, N., Narain, R., and Lin, M. C., "Efficient and accurate sound propagation using adaptive rectangular decomposition," volume 15, pp. 789–801, 2009.

[32] Cuevas-Rodriguez, M., Gonzalez-Toledo, D., Rubia-Cuestas, E., Garre, C., Molina-Tanco, L., Reyes-Lecuona, A., Poirier-Quinot, D., Picinali, L., Molina, L., and de La Rubia-Cuestas, "An open-source audio renderer for 3D audio with hearing loss and hearing aid simulations," in *142nd AES Convention*, 2017.

[33] Cuevas-Rodriguez, M., Gonzalez-Toledo, D., Rubia-Cuestas, E. D. L., Garre, C., Molina-Tanco, L., Reyes-Lecuona, A., Poirier-Quinot, D., and Picinali, L., "The 3D Tune-In Toolkit - 3D audio spatialiser, hearing loss and hearing aid simulations," pp. 1–3, 2018.

[34] Rosen, M., Godin, K. W., and Raghuvanshi, N., "Interactive sound propagation for dynamic scenes using 2D wave simulation," *Comput. Graphics Forum*, pp. 39–46, 2020.

[35] Poirier-Quinot, D., Katz, B. F. G., and Noisternig, M., "EVERTims: Open source framework for real-time auralization in architectural acoustics and virtual reality," 2017.

[36] Noisternig, M., Katz, B. F. G., Siltanen,

S., and Savioja, L., "Framework for Real-Time Auralization in Architectural Acoustics," *Acta Acustica united with Acustica*, pp. 1000–1015, 2008.

[37] Grimm, G., Luberadzka, J., and Hohmann, V., "A toolbox for rendering virtual acoustic environments in the context of audiology," *Acta Acustica united with Acustica*, pp. 566–578, 2019.

[38] "Virtual Acoustics," 2017, `https://www.virtualacoustics.org/`.

[39] Haas, K. D., Schutte, M., and Ewert, S., "Real-time Virtual Environment and Room Acoustics Simulator," in *Forum Acusticum Euronoise*, 2025.

[40] Wendt, T., van de Par, S., and Ewert, S., "A Computationally-Efficient and Perceptually-Plausible Algorithm for Binaural Room Impulse Response Simulation," *J. Audio Eng. Soc.*, pp. 748–766, 2014.

[41] Schröder, D. and Lentz, T., "Real-Time Processing of Image Sources Using Binary Space Partitioning," *J. Audio Eng. Soc.*, pp. 604–619, 2006.

[42] Möller, T. and Trumbore, B., "Fast, minimum storage ray/triangle intersection," in *ACM SIGGRAPH*, Association for Computing Machinery, Inc, 2005.

[43] Svensson, U. P., Fred, R. I., and Vanderkooy, J., "An analytic secondary source model of edge diffraction impulse responses," *J. Acoust. Soc. Amer.*, pp. 2331–2344, 1999.

[44] Mannall, J., Savioja, L., Calamia, P. T., Mason, R., and Sena, E. D., "Efficient Diffraction Modeling Using Neural Networks and Infinite Impulse Response Filters," *J. Audio Eng. Soc.*, pp. 566–576, 2023.

[45] Kirsch, C. and Ewert, S. D., "A Universal Filter Approximation of Edge Diffraction for Geometrical Acoustics," *IEEE/ACM Trans. Audio, Speech, Language Proc.*, pp. 1636–1651, 2023.

[46] Grimm, G., Wendt, T., Hohmann, V., and Ewert, S. D., "Implementation and Perceptual Evaluation of a Simulation Method for Coupled Rooms in Higher Order Ambisonics," pp. 27–32, 2014.

[47] Oliver, R. J. and Jot, J.-M., "Efficient Multi-Band Digital Audio Graphic Equalizer with Accurate Frequency Response Control," in *139th AES Convention*, Audio Engineering Society, 2015.

[48] Mannall, J., Calamia, P. T., Savioja, L., Neidhardt, A., Mason, R., and Sena, E. D., "Assessing Diffraction Perception Under Reverberant Conditions in Virtual Reality," Audio Engineering Society, 2024.

[49] Brinkmann, F., Aspöck, L., Ackermann, D., Opdam, R., Vorländer, M., and Weinzierl, S., "A benchmark for room acoustical simulation. Concept and database," *Appl. Acoustics*, 2021.

[50] Nwobodo, O. J., Wereszczynski, K., and Cyran, K., "A review on tracking head movement in augmented reality systems," *Procedia Computer Science*, pp. 4344–4353, 2023.

[51] Broderick, J., Duggan, J., and Redfern, S., "The Importance of Spatial Audio in Modern Games and Virtual Environments," pp. 507–510, 2018.

Model	Access	Interface	Spatialisation	Early Reflections	Diffraction	Late Reverberation	Room Geometry
3DTI Toolbox [32]	Open-source (code available)	C++, Unity, java, VST, Standalone app	Custom HRTF	Binaural room impulse responses	Binaural room impulse responses	Binaural room impulse responses	Any measured room
Planeverb [34]	Open-source (code available)	C++, Unity	VBAP	Canonical filters parameterised from 2D FDTD model	Obstruction gain parameterised from 2D FDTD model	Canonical filters parameterised from 2D FDTD model	General geometry suitable as a 2D slice
EVERTims [35]	Open-source (code available)	C++, Blender	Ambisonics	Beam tracing model	✗	FDN parameterised using Sabine's equation	General geometry (excluding double slope decay)
TASCAR [37]	Open source (code available)	Jack Audio Connection Kit, MATLAB	Custom HRTF	Image-source model	✗	FDN parameterised from room geometry	General geometry (excluding double slope decay)
Audiokinetic: Reflect [9] and RoomVerb [30]	Paid license	Wwise	HRTF (using Auro headphone)	Image-edge model	LPF and gain based on bending angle	Manually tuned and coupling between rooms	General geometry
Meta XR Audio SDK [8]	Free to use	Unity, Unreal, Wwise, FMOD	HRTF	Early reflections (up to 12)	Shadow zone only, one path per source	Stochastic parametrised from ray tracing	General geometry
Project Acoustics [10]	Microsoft Game Studios	Unity, Unreal, Wwise	HRTF	Canonical filters parameterised from ARD model	Obstruction gain parameterised from ARD model	Canonical filters parameterised from ARD model	General geometry
RAVEN [4]	Academic use	MATLAB, SketchUp, Virtual Acoustics	Custom HRTF	Image-source model	✗	Stochastic parametrised from ray tracing	General geometry
liveRAZR [39]	Available on request	OSC messages	Custom HRTF	Image-edge model	UDFA (Shadow zone only, one path per image source)	FDN paramaterised from room geometry	General geometry (excluding double slope decay)
RoomAcoustiC++ (ours)	Open-source (code available)	C++, Unity	Custom HRTF	Image-edge model	BTM, UTD, NN-IIR, UDFA models	FDN parameterised from room geometry	General geometry (excluding double slope decay)

Table 1: Comparison of existing real-time room acoustic modelling tools and the proposed RoomAcoustiC++ library. Colour coding criteria is based on (a) Open source, (b) Binaural spatialisation including custom HRTFs, (c) Geometry based early reflections, diffraction and late reverberation. Colour coding: green (north east hatching) fulfils criteria, orange (grid hatching) partially fulfils criteria, red (north west hatching) does not fulfil criteria. Used abbreviations: head-related transfer function (HRTF), vector based amplitude panning (VBAP), finite-difference time-domain (FDTD), adaptive rectangular decomposition (ARD), feedback delay network (FDN), low-pass filter (LPF) Biot-Tolstoy-Medwin (BTM), uniform theory of diffraction (UTD), Universal Diffraction Filter Approximation (UDFA)