Rendering environmental voice reverberation for large-scale distributed virtual worlds

Micah Taylor*
University of North Carolina

Nicolas Tsingos†
Dolby Laboratories

Dinesh Manocha‡
University of North Carolina

Abstract

We propose an approach that can render environmental audio effects for a large number of concurrent voice users immersed in a large distributed virtual world. In an off-line step, our approach efficiently computes acoustic similarity measures based on average path length, reflection direction and diffusion throughout the environment. The similarity measures are used to adaptively decompose the scene into acoustic regions Sound propagation simulation is performed on the acoustic regions, resulting in acoustic response data that can be used efficiently at runtime along with deferred late reverberation. We demonstrate realtime realistic sound rendering of large number of voice streams in virtual environments of tens of square kilometers of area at a fractionof the authoring and memory cost of previous acoustical precomputation approaches.

CR Categories: H.5.5 [Information Interfaces and Presentation]: Sound and Music Computing—[modeling, systems]

Keywords: radiosity, global illumination, constant time

Links: DL PDF

1 Introduction

For networked virtual environments, such as social communities or massively multiplayer on-line (MMO) games, meaningful interaction through voice conversation with other participants is a valuable feature [Williams et al. 2007; Wadley et al. 2007; Sallnäs 2005]. First adopted through side-clients enabling telephone quality, walkie-talkie style communication, voice services are becoming more integrated and are now connecting hundreds of millions of users on PCs, game consoles and cell phones. For instance, group voice chat is integral to gaming services such as Microsoft Xbox LIVE, Sony Playstation Network, and Valve Steam and is also directly integrated into such games as Blizzard's World of Warcraft, CCP Games' EVE Online, Electronic Arts's Need for Speed World and Linden Lab's Second Life.

There has been much work on voice communication over the internet, typically called VoIP (Voice over Internet Protocol) [Goode 2002]. Latency, voice coding efficiency and network error resilience as well as endpoint voice cleaning and processing are some

*e-mail: taylormt@cs.unc.edu
†e-mail: nicolas.tsingos@dolby.com
‡email: dm@cs.unc.edu

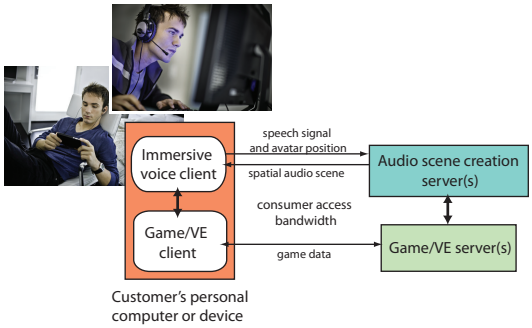


Figure 1: Overview of an immersive and scalable VoIP system: *a voice server is responsible for computing a spatial auditory scene for each connected client, mixing or forwarding the voice streams based on the local density of active users.*

of the key issues [Markopoulou et al. 2002; Benesty et al. 2000]. While most work in scalable VoIP focuses on the infrastructure system, there has been limited research on improving the immersive effects of the voice communication, typically through spatialized rendering [West et al. 1992; Hollier et al. 1997]. Studies have shown [Halloran 2009] that while voice communication helps users coordinate in virtual environments, the lack of environmental effects can cause difficulty in identifying sound sources. Modeling the effects of sound propagation such as occlusion and echoes can help convey scenes where participants communicate from different rooms or areas. For example, the direction of the sound reflections can help a user spatialize the sound source position, while the time it takes for the echoes to decay conveys the size of the environment. The direct sound path and early sound reflection help the listener spatialize the sound source, while the late reverberation echoes convey the scale of the environment and the materials present. However, these acoustic effects are difficult to implement in a large-scale VoIP system. Since the voice mixing is generally performed on a remote server, network delivery cost restricts the amount of data that can be transferred. Moreover, a typical 8-core server platform must handle thousands of remote clients simultaneously, strongly limiting the processing capabilities. As a result, the most advanced VoIP systems currently implement direct line-of-sight occlusion modeling as well as simplified diffraction effects resulting in unrealistic proximity cues. For MMO games where localizing teammates and enemies is of primary importance, rendering inappropriate distance cues can lead to a tactical disadvantage.

In this paper, we focus on the immersive rendering of environmental reverberation effects on voice streams for such large-scale distributed virtual worlds.

1.1 Scalable spatial voice

The Massive system was an early project to enable voice communication in immersive environments [Greenhalgh and Benford 1995]. As voice communication is heavily used in MMO gaming, there is a rising need for wider scale voice communication. Much work has been done on creating methods for handling these large environments [Radenkovic et al. 2002; Boustead and Safaei 2004; Safaei

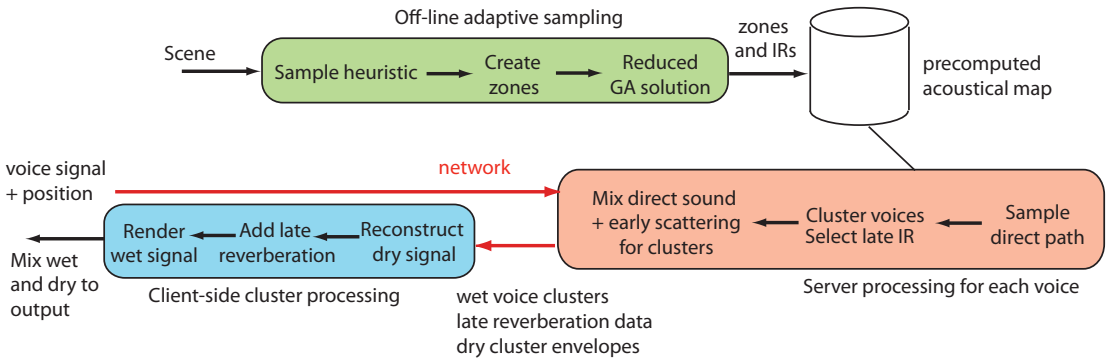


Figure 2: Example integration into a VoIP system: *Our efficient pre-computation and deferred reverberation algorithm enable realtime rendering of environmental audio effects in large-scale environments with many connected clients.*

2005].

In these games, a player can be in audible range of many other players which requires the voice streams to be routed through a server that dynamically optimizes the auditory scene. In order to scale to large numbers of users, VoIP servers generally implement a combination of voice-packet forwarding as well as mixing of the voice streams on the server (Figure 1). In the mixing mode, the server creates a simplified representation of the voice scene audible to each client by clustering, that is, grouping different voice streams together into a small number of combined streams. The audio mixture corresponding to all the voice streams grouped in a cluster is computed on the server and streamed back to the client. The number of clusters, typically between one and three, is significantly lower than the number of connected clients therefore limiting the required bandwidth. In most available commercial solutions, the bandwidth is kept between 16 and 48 kilo-bits per second (kbps). The cluster signals are delivered to the client with spatialization information (e.g., direction of incidence) so that they can be rendered over surround loudspeakers or binauralized over headphones [Begault 1994]. If the number of active talkers is small, the voice streams can be directly forwarded to each client, avoiding a decode/mixing/re-encode step. In this case, any further processing must be carried out client-side and the server needs to forward all the required metadata.

1.2 Sound reverberation and environmental effects

In current video games, reverberation is either directly pre-rendered into the sound effects or implemented at run-time using dynamic artificial reverberation filters [Jot 1999]. Parameters of the reverberation decay can be directly manipulated by the sound designer to achieve a desired effect without requiring any geometrical modeling. While simplifying the authoring process, traditional artificial reverberators suffer from a number of issues. They impose a "single room" model and constrain the shape of the decay profile (e.g., exponential). These methods also make limited use of geometry and therefore fail to convincingly model coupled or outdoor spaces. In [Bailey and Brumitt 2010], distance histograms extracted from a cube map rendered at a position of interest are used to select the parameters for such an artifical reveberator. However distances to the camera origin alone fail to capture local variations of the surfaces which have a strong influence on the scattering properties [Tsingos et al. 2007].

Client-server solutions have been proposed in the past to dynamically compute sound propagation paths between clients connected in a virtual environments using the actual geometry [Funkhouser et al. 1999]. But even the most recent geometrical acoustic (GA) approaches that can model dynamic sound reflection and diffraction interactively [Taylor et al. 2009] cannot scale to large environments.

A practical approach to simulating acoustics of virtual environments is to pre-compute the acoustical impulse response at several locations throughout the environment in an off-line process so that the results can be efficiently re-used to process the audio signals at run-time [Pope et al. 1999; Tsingos 2009; Siltanen et al. 2009; Raghuvanshi et al. 2010]. The main benefit of the off-line computation is that both early and high-order sound scattering (reflection/diffraction) can be simulated, providing improved proximity cues and distance perception. However, due to the high precomputing cost and the difficulty of combining reverberation processing with clustering or spatial scene simplification, thus far, none of the previous work was able to render convincing early sound scattering and reverberation for large numbers of participants in large-scale virtual environments.

1.3 Overview

To address this challenge, we introduce three main contributions:

1. **Acoustical similarity measure:** We introduce a geometric measure which correlates well with variations in the acoustic field as predicted by acoustic simulations. The measure can be computed using the local neighborhood of a given point location in the environment.

2. **Acoustic region generation:** We use the similarity measure to sample the virtual environment and then group similar samples into acoustic regions. Since regions are segmented by acoustic properties, our system need only sample the full acoustic responses in each region. This results in a reduction of both the precomputation time cost and the cost of storing the precomputed data, enabling us to handle very large scenes, spanning kilometers in virtual space.

3. **Deferred late reverberation:** We introduce a solution to select a single late reverberation filter for a group of sources while preserving individual early reflection processing. The early reverberation is processed by the server for each voice while late reverberation effects are deferred to the client. This makes our approach compatible with sound source clustering.

Figure 2 offers an overview of the proposed approach in the context of a large-scale VoIP system as used in multi-player online games. Our new algorithms can be used to automatically create environmental reverberation maps up to 5 times faster than previous solutions on indoor scenes and up to 80 times faster on outdoor scenes. In addition, memory costs are also reduced by similar factor and our optimized deferred processing allows rendering of reverberation effects for scenes comprising thousands of connected clients.

The rest of this paper is organized as follows. Section 2 introduces our acoustical similarity measure. Section 3 details our adaptive refinement precomputation and section **??** our run-time deferred reverberation method. Sections 4 and 5 provide an analysis of the accuracy and performance of our methods.

2 Acoustic Response Similarity

In this section, we introduce an acoustic response similarity measure that is used for offline adaptive sampling. We desire our measure provides a good estimate of the acoustic properties and response at a point in the scene. Furthermore, it is important that we can compute this measure efficiently.

An acoustic response is computed between a source point and a receiver point. However, given n possible sampling points in the scene, there are $O(n^2)$ possible acoustic responses. We observe that for a given source location, the geometric configuration of the scene walls and objects directly controls the acoustic response at any receiver. Many geometric acoustic methods, such as the ones based on beam tracing [Funkhouser et al. 1998] and frustum-tracing [Chandak et al. 2009] take advantage of the static nature of the scene in terms of precomputing the visibility tree. These methods assume that there is either a single fixed source or the receiver's position is fixed. For any receiver position, the visibility tree can accurately predict the acoustic response for up to the number of orders of response that the tree was precomputed. However, the tree computation is relatively expensive a separate tree is need for each source position. Moreover, in our case we would like to develop an approach that is applicable to large indoor and outdoor scenes.

We present a similarity measure that uses low order sampling of the geometric primitives in the scene can be computed efficiently using standard visual algorithms such as *cube mapping* [Greene 1986]. Our method retains the advantage of being independent of receiver position. To compute our measure, we sample the geometric properties that influence high order reflections: surface distance and surface normal. Since our measurement requires spatial data from the scene, we use cardinal axes aligned cube maps.

Once the surface distance and normal are computed, a gradient is taken over surface distance and surface normal. The three values corresponding to distance, normal, and gradient relate to the physical properties that influence the reflection of sound in the scene: reflection distance, reflection direction, and reflection diffusion, respectively.

These data values are stored per each face of the cube map and forms a first order response measure related to the surfaces near the sampling point. Storing the cube face images for each sample point requires large amounts of storage and can result in very high storage overhead for large scenes. To avoid exhausting memory when rendering large scenes, an integration step is used to compute the mean of the geometric properties on each cube face. Given a geometric value v (representing one of the three data values) across s cube map samples, we compute the mean v_{avg}:

$$v_{avg} = \sum_{i=0}^{s} \frac{v_i}{s}.$$

The spatial data we measure corresponds to some of the physical properties that influence the first order of sound reflections: average path length, average reflection direction, and average diffusion.

2.1 Path length

We assume it is likely that most first order acoustic paths will travel by reflection off a nearby object. After reflection, the sound waves propagate to the receiver position. The acoustic path may then be viewed as a combination of two segments: the first segment is the path from the source to the reflecting object, and the second segment is the path from the reflecting object to the receiver.

Given an object at distance d from the source position, and a receiver at distance r from the source position, the longest length t of the first order reflection path occurs when the object and receiver are on opposite sides of the source, such that the reflection path passes through the source position before arriving at the receiver, resulting in a path length of ℓ:

$$\ell \leq d + (d + r)$$

The shortest reflection path occurs when the object lies near planar to the path between source and receiver, resulting in a path length of nearly $d + (r - d)$ with some additional distance caused by perpendicular deviation distance e from the straight line direct path. The shortest reflection path length is given as:

$$\ell \geq d + \sqrt{e^2 + (r - \sqrt{d^2 - e^2})^2}.$$

As e goes to 0, the reflection path becomes the direct path. We note that for both the longest and shortest path cases, there is a distance term which is directly related to the distance d to nearby objects, and a reflection term of d and r are based on the receiver's geometric relation to d. All reflection paths are variants of these terms.

In our similarity measure, we do not assume a given receiver position, so our measure is based only on object distances. This average distance value serves at the first component in our similarity metric. As the distance measure changes, the average first order reflected path length is also likely to change.

2.2 Reflection direction

The incoming direction of the earliest sound paths to a receiver is highly indicative of the direction of the sound source. In cases when there is no line of sight between the source and the receiver, this cue is especially important. The direction of an object's surface normal directly influences the direction of any reflected sound paths off the surface.

Reflection direction $\mathbf{r}$ can be determined give the view direction $\mathbf{v}$ and the surface normal direction $\mathbf{n}$. We note that $\mathbf{v}$ is fixed by the cube map sample location, and the direction is a function of $\mathbf{n}$, given as:

$$\mathbf{r} = 2(\mathbf{v} \cdot \mathbf{n})\mathbf{n} - \mathbf{v}.$$

Since the surface normal directly influences the direction of first order reflection direction, we use the normal as the second component in our measure.

2.3 Diffuse scattering

The amount of diffuse scattering of surfaces can significantly influence the final acoustic response at the receiver's location. As

such, source positions that are likely to have very different specular/diffuse properties are likely to produce very different sound fields.

We measure the surface gradient with respect to surface normal and surface depth to estimate diffuse reflectance properties. For a given cube map face the depth values and surface normal values are known. We then use a series of 2D operators over the cube face to find discontinuities in depth gradient and normal gradient. These discontinuities represent regions where diffraction is likely and diffusion will be introduced into the sound field.

First, a 2D gradient is computed from the depth information:

$$\nabla_x = [1, -1]; \quad \nabla_y = \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

This results in a two component image representing depth changes in x and y directions. To estimate portions where scattering is likely to occur, we detect discontinuities in this image. An edge detection kernel is applied to the components of the gradient:

$$k = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}.$$

The maximum value of the two components is retained and clamped to the range $[0, 1]$, resulting in a black and white map with depth and normal discontinuities detected. Since the depth and normal discontinuities influence the scattering of sound in the scene, we use these discontinuities as the final component in our measure.

2.4 Similarity comparison

In the previous sections, we described our method for extracting geometric properties that correspond to expected reflection path length, expected reflection direction, and expected diffuse energy. If two sample points have similar values for these parameters, it is likely that the acoustic field measured around the sample positions will be similar. The similarity measure S between sample points $\mathbf{a}$ and $\mathbf{b}$ can be computed using the three metric previously described:

$$\mathbf{S} = < d^{\mathbf{a}}_{avg}, r^{\mathbf{a}}_{avg}, f^{\mathbf{a}} > - < d^{\mathbf{b}}_{avg}, r^{\mathbf{b}}_{avg}, f^{\mathbf{b}} >; \quad S = ||\mathbf{S}||,$$

where d_{avg} is the mean surface distance, r_{avg} is the mean surface normal, and f is the estimated diffusion coefficient.

If each component of this measure is below defined threshold values, the two sample points are considered similar. In addition to these parameters, line of sight is also used to restrict similarity. For two sample points to be considered similar, there must exist a line of sight between the points.

This forms the basis for our similarity metric: sample points are likely to have a similar acoustic response if the early response (cube map measured data) is similar and the points are visible to each other (LOS restriction). Appropriate similarity threshold values are shown in Section 4.

3 Scene Decomposition and Sampling

Precomputing the acoustic response in large scenes can be challenging. A common method is to compute and store the acoustic field for many possible listener and source positions. This can be performed by placing a series of sample points in the scene, and measuring the acoustic response between each pair of sample points. This is an $O(n^2)$ computation, where n is the number of positions and can result in high computational overhead and storage cost, even for small scenes. However, this approach has a relatively small runtime overhead as the resulting data can be queried based on the sample points that are closest to the source and receivers. In this section, we present a scene decomposition algorithm that reduces the time and storage cost of the acoustic precomputation, while maintaining the same level of accuracy.

Previous approaches either precompute the environment's acoustic response over a densely sampled regular grid or generate a single acoustic response based on arbitrary decompositions of acoustic space (often individual rooms) [Raghuvanshi et al. 2010]. One possibility is to use a regularly sampled, dense grid which can approximate the sound field with high accuracy. However, a dense grid requires large amounts of storage and computation time. Additionally, in large open scenes, dense sampling may be overly conservative, as there will be little variation between nearby samples. Conversely, other methods that precompute a single response per acoustic space are more efficient in terms of space and computation time, the sampling resolution may be too low to capture directional effects such as early reflections.

Precomputation storage cost can also be lowered by reducing the number of samples needed. This can be achieved by decreasing the sample density or by combining nearby sample points that have similar acoustic impulse responses. Unfortunately, combining sample points in this manner only reduces the final *storage cost*, not the *time cost* required to precompute the scene response. This is due to the fact that checking if two sample points have similar acoustic responses requires the response to already be known at each point. Our goal is to reduce the precomputation time as well as storag overhead.

The storage cost of final acoustic field data is reduced by eliminating acoustic sampling at positions that are likely to be very similar to a nearby sample. The time cost of computing the acoustic response is reduced by using by using our similarity metric instead of performing a complete uniform sampling of the acoustic space. We note that our similarity measure is based on surface properties that contribute to the first order acoustic response and not the actual acoustic response.

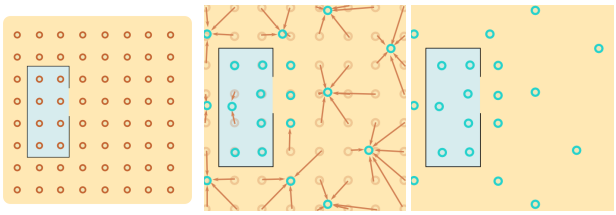
Moreover, since the similarity measurement is computed using the surfaces of the objects near the sample point, this measurement is not directly related to an individual receiver position. This is in contrast to a typical acoustic response computation that represents a sampling of the acoustic field between a source and receiver. Our similarity measurements are computed in $O(n)$ time.

Since the similarity measure is fast to compute, our algorithm first densely samples the similarity measure on a grid of points. Once the surface properties near each sample point have been measured, sample points with similar properties are merged and the total sample point count in the scene is reduced.

3.1 Sample selection

Once the similarity properties at each sample point have been measured, the scene decomposition algorithm is used to compute adaptive sampling. We note that the similarity measure S is useful for evaluating whether two sample positions are similar, but not directly useful for eliminating the sample positions.

If a sample point is removed, the nearby acoustic field will be sampled more sparsely and this may lead to more error at reconstruc-



(a) (b) (c)

Figure 3: Adaptive sampling: *(a) Regular grid sampling creates a very high number of samples while adaptive refinement (b) removes redundant samples resulting in an adaptive sampling of the scene with fewer samples (c).*

tion when a receiver that is placed near that sample point. To reduce the error in the final response computation and to determine which sample positions should be merged and eliminated, the error cost of removing each sample point is computed. If a given sample point is merged with a neighboring point, there is a single resulting sample point that serves as a representation of the original sample positions.

We create a graph of sample points with edge nodes that connect neighboring sample points. Edge weights are determined by the error in similarity measure that will result if the sample points are merged. For two sample points with similarity measures S_1 and S_2 and top, bottom, left, and right edges over four dimensions (with *dim* representing the direction corresponding to each neighboring similarity measure), the merge cost C can be given as:

$$S_a = \frac{S_1 + S_2}{2}$$

$$C = \sum_{dim=0}^{4} (S_a - S_1^{dim}) + \sum_{dim=0}^{4} (S_a - S_2^{dim})$$

These sample points are merged with their neighbors in a greedy manner. In order to avoid exceeding the error thresholds in the similarity measure, the error in merging two points p is compared against the similarity threshold S_{thr}:

$$\max_{dim,p} (S_a - S_p^{dim}) < S_{thr}$$

If the error criteria is satisfied, these points may be merged to compute a new sample point. The new sample point position and similarity measure are defined as the average position and average similarity measure of the combined points. As nearby sample points are merged, the merged edges are removed from the graph and the costs associated with all the edges incident to those sample points are reevaluated. The process repeats until all points have been combined or fail to satisfy the error threshold.

3.2 Acoustic regions

The output of the refinement stage is an irregular arrangement of samples (see Figure 3). However, the process of sample merging forms the acoustic regions. If two sample points were merged, they were determined to be similar by the error criteria. If two points are merged, their original positions in the 2D grid are marked as belonging to the same region. As points are merged, the acoustic regions grow until the edges at the boundary of the region no longer satisfy the error criteria.

At runtime, the acoustic region of the source point and receiver point must be selected. This is based on the spatial position of the source and receiver. In general environments, the source and receiver will not lie directly on the sample points used during precomputation.

The acoustic region of the source and receiver is computed by checking if the four nearest grid locations. The simplest case is when the four nearest grid locations were combined into a single acoustic region. This region can then be immediately determined. If not, line-of-sight queries with $O(\log n)$ time cost are conducted to the four points against the objects in the scene. The closest point with clear line-of-sight is selected as the closest grid node.

3.3 Simulation of sound propagation

Our method treats the sound propagation simulation as a black box and has no requirements other than scene input and impulse response output formats. We have chosen to use a GA simulation in order to compute responses on scenes of very large size.

Our simulator traces specular rays [Vorländer 1989]. Given a receiver position and a source position, our simulator outputs an acoustic impulse response that represents how the environment affects the sound waves that travel from the source to the receiver. This impulse response can be convolved with any input signal to render an output signal with the appropriate effects.

We consider each sample obtained after the previous decomposition process as a source and trace rays from the position, collecting propagation paths to every other sample, which act as listeners.

3.4 Response storage

Reverberation filters can be represented in a compact manner by sampling the energy decay profile through time [Merimaa and Pullki 2004; Tsingos 2009]. The decay profiles are built by integrating the energy in the impulse response over small time-steps and a number of frequency sub-bands. In this way, both temporal and frequency resolution can be controlled by the user. Similar to previous work [Tsingos 2009], we store pressure values for several spectral sub-bands quantized in the time domain. In addition to the spectrum data, directionality and diffusion data is stored. An energy-weighted average direction of incidence is stored at time-step resolution, with all frequency bands sharing the same direction. Similarly, a directional-to-diffuse energy value is also computed [Merimaa and Pullki 2004].

Each impulse response signal is split into an early and late response component. The boundary can be controlled by the user. The pressure values in the late portion of the response are quantized in log scale to the nearest value and normalized to the lowest value. The normalization factor is recorded during this process. The three parts of this operation, early response, late response, and normalization factor are stored in a bank of reverberation data.

The normalized late response is compared against all previously collected late response data in the late response data structure. If the response difference is less than a user specified error threshold, the mean of the responses is stored in the bank, and the new response data is discarded. Early responses and the normalization factors are not combined when inserting into the response bank.

3.5 Response reconstruction

Since the paths that are captured in the early response portion are dependent on the source and receiver positions, large acoustic regions produce overly strong early response output when source and

receiver have a large separation distance, but are in the same region. When reconstructing the final signal, the early response portion has a distance attenuation factor applied to it. We scale each pressure value by $\frac{1}{d}$, where d is the distance that leads to to a response in that time step. This attenuation factor is reduced over time, with no attenuation be applied to the final early time sample. This process allows the early field to be attenuated, while not altering the standing late reverberation field.

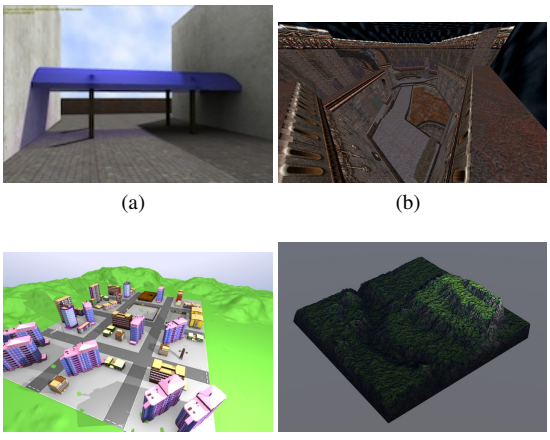
Final reconstruction combines the early response with the late response. The late response is scaled by its normalization factor and appended to the early response. The performance of our reconstruction method is analyzed in section **??**.

4 Analysis

In this section, we discuss the results generated by our algorithms. We analyze the complexity of our approach and the accuracy of our precomputation method as well as the quality of the defered late reverberation processing. For analysis, we have used several benchmark scenes shown in Figure 8.1. The selected test scenes represent likely use cases, with both indoor and outdoor scenes corresponding to game maps and virtual worlds. The scene details are described in Table 1.

Scene	# Triangle	Size (m)	Grid spacing	Grid count
Simple outdoor	2k	33 x 33 x 10	2m	289
FPS game	14k	30 x 60 x 20	2m	465
Small city	2k	245 x 310 x 33	2m	800
Large city	4k	600 x 980 x 33	2m	132k
Canyon	540k	4k x 4k x 100	10m	160k

Table 1: Benchmark scenes: *These scenes are used throughout the analysis section for accuracy and performance metrics.*



(a) (b) (c) (d)

Figure 4: Benchmark scenes: *We illustrate the benchmark scenes to highlight the performance and accuracy of our algorithm: a) Simple outdoor, b) FPS game, c) City (Small city is a subsection of this scene), d) Canyon.*

4.1 Similarity measure thresholds

During the precomputation step, our method computes an acoustic similarity measure for many positions in the environment. An error threshold is used to select regions that are similar when decomposing the scene.

When merging sample points and decomposing the scene, the error between to sample points then ranges between $(0, 1]$. For the results presented in this paper, we set our error threshold S_{the} to 0.3. When combining the portions of the late response, we divide early and late after 640 milliseconds. At a sampling rate of 16kHz, this results in an early response of 32 time steps of 20 milliseconds and a late response of 368 time steps.

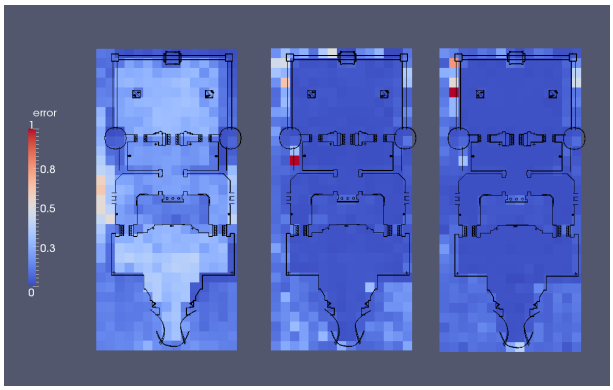
4.2 Error computation

We measure the error in the reconstructed results based on properties of the impulse responses. We compare the initial onset time, average early direction, and the reverberation times in the form of RT60. The ground truth data is the full set of responses from a GA simulator with source and receivers set at the same grid size. In the ground truth and our decomposed simulation, we traced 50k rays for 50 reflections for each sample point.

Due to the decomposition process, it is possible for some regions to have different responses from the reference GA solution. Moreover, some positions in one solution may have response values, while the same positions in the other solution have no values. In this case, the error at those positions is undefined and we assume the worst possible error value at these locations. For initial onset, a missing value indicates the onset occurred outside of the measurement range, resulting in temporal error of 100%. For early direction, when values are not present, we assume a maximum error of π radians from the ground truth. RT60 is estimated by a least squares fit in log space, and is assumed to be zero when no value is present.

4.3 Precomputed response accuracy

Since our system reduces the number of acoustic responses by based on our similarity measure, it is possible that the final acoustic map does not accurately represent the environment. We compare the results for the sample points from our precomputation to a complete GA simulation for a 2m x 2m grid of sample points for the same environment. Figure 5 shows the error visually.



(a)

Figure 5: Response accuracy: *This figures shows the expected error from a source position to any receiver in the Game FPS scene. A wireframe of the scene is presented at the top (a). Red areas indicate high error. From left to right the error plots are: early direction error, initial onset error, and RT60 error.*

By adjusting the error threshold in the region segmentation step, higher reconstruction accuracy can be achieved. This requires more acoustic regions to be stored. Figure 6 shows how our method can have high accuracy with a small number of samples.

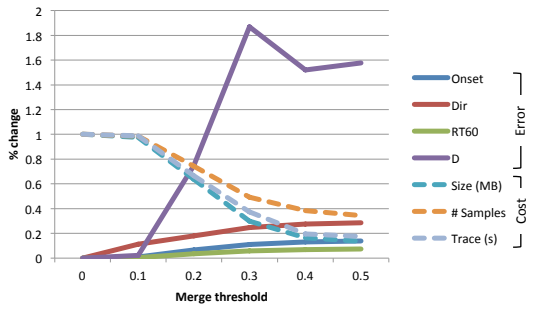


Figure 6: Sampling accuracy: *As merge threshold error is reduced, error is decreased at the expense of storage and time cost.*

.

5 Performance

In our tests we used an NVIDIA 480 GTX and an Intel Xeon 5150 at 2.66 GHz. The test GPU has 1.5GB memory and the test CPU has 4GB memory. The cube sampler and similarity metrics were implemented in OpenGL with GLSL shaders to provide the diffusion and integration steps. The reduction process, the pre computation, and the analysis programs were implemented in C++ with OpenMP threading.

5.1 Similarity and reduction cost

Our similarity metrics are compute on cube faces using GPU hardware. Custom GLSL shaders sample the surface distance and normals. Distance is computed in world space scales and world normals are recorded. The diffusion metric is computed in one pass by computing the gradient for each query in the edge convolution kernel. Integration is a simple texture value summation kernel, scaled by the kernel size. We found that a kernel size of 4 pixels was optimal for our hardware.

The reduction step line-of-sight queries are computed using a fast CPU based BVH ray tracer. All edge reduction and region computations are done using custom data structures backed by STL containers. The reduction process is not multi-threaded due to its already low overhead compared to the other tasks.

Using the error thresholds described in the previous section, we preformed sampling and segmentation on the benchmark scenes. The time to compute the similarity measure at each sample point, as well as the time cost to eliminate similar sample points is shown in Table 2.

Scene	Full grid	Reduced			Time improvement
		Cubemap	Segmentation	Trace	
Simple outdoor	146.8s	1.3s	1ms	–	–
FPS game	35m	2.0s	1ms	12m	2.9x
Small city	12.3h	8.6s	3ms	2.9h	4.24x
Large city	192h*	26s	69ms	16.3h	11.8x
Canyon	13d*	5.2m	22s	37.8h	8.25x

Table 2: Precompute time cost: *Region segmentation using cube maps allows a significant reduction in precompute time. Full grid computes marked with a * were estimated by subsampling a number of grid positions and scaling by the number of remaining grid positions.*

5.2 Precomputation

By segmenting the scene and reducing the number of samples points used in acoustic precomputation, the time and storage cost to simulate the acoustic response across a virtual environment can be greatly reduced.

Our sound propagation simulator is based on discrete ray tracing, accelerated by efficient BVH trees. The ray tracer is heavily multi-threaded, but not SIMD accelerated due to the diffusion that occurs at high reflection orders. Receivers are modeled as spheres and intersection paths are not validated as occlusion free. The simulator computes the response at each receiver in the scene from a single source in one simulation cycle. This results in n simulation cycles for n regions or source positions.

We store our acoustic data as quantized decay filters. A four second reverberation decay profile can be efficiently encoded using 200 blocks each containing 6 bytes of information for a total of 1.2 KB [Tsingos 2009]. Reducing sample count also significantly reduces the memory cost compared to grid methods. Table 3 compares our storage method to standard grid based methods.

Scene	Full		Reduced		Storage improvement
	samples	storage	samples	storage	
Simple outdoor	–	–	–	–	
FPS game	216k	259MB	36k	81MB	3.19x
Small city	640k	768MB	–	–	–
Large city	132k	20TB	9.4k	2186MB	9,149x
Canyon	160k	30.7TB	35k	2316MB	13,347x

Table 3: Storage cost: *Our storage algorithm reduces nearly intractable scene storage to a manageable size.*

5.3 Runtime cost

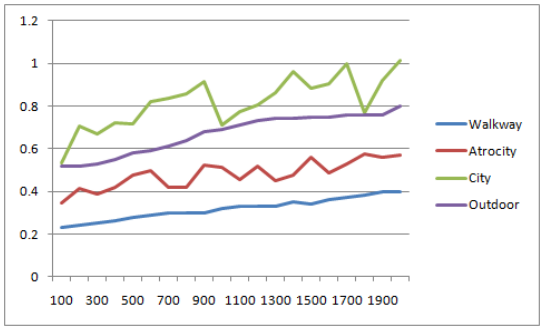
By implementing our runtime mixing in efficient GPU kernels, we can mix thousands of clusters in realtime on high-end GPUs. We implemented CPU and GPU versions of our runtime mixer. Results are presented in Table 4.

Scene	10 clients (ms)	100 clients (ms)	1000 clients (ms)
Walkway	–	–	–
Game FPS	6	63	–
City	6	65	–
Large outdoor	7	68	–

Table 4: Runtime time costs: *Even when simulating many sound sources, our system can mix hundreds of audio streams in realtime. We assume that all clients have 3 audible streams to be mixed.*

In traditional spatial VoIP systems, occluded voices not in direct line of sight would fade out to silence and therefore would not be considered for mixing, which can indirectly improve the compute and bandwidth scalability of the system. In our case, occluded voices will still be considered for clustering as they might be audible due to early scattering and reverberation which will decrease the forwarding-to-mixing ratio and therefore require more processing.

We measure performance by the number of audio streams that can be rendered simultaneously. For our purposes, an audio stream is the data that must be mixed for each source to each receiver. For example, if a single source is within range of three receivers, three streams would need mixed. Details on how our method scales with streams is shown in figure 7.



(a)

Figure 7: Runtime scaling: *Implementations of our method can scale up to thousands of sources while rendering all streams in realtime (i.e. less than 100% processor time used).*

6 Discussion and limitations

6.1 Comparison

Other pre computation methods have been presented. However, none focus on fast rendering and automatic scene decomposition, both needed when rendering large scenes with many sources. We compare our approach to other pre computation methods in Figure 8.

Algorithm	Ours	PART	Wave grid	IS gradiant	Dir. Prop. Cache	Reverb graph
Memory requirements	Low	Low	High	Low	Medium	Low
Convolution	>Realtime	Interactive	Realtime	>Realtime	Realtime	Realtime
Regions	Acoustical	Spatial	None	Cell+Portal	Cell+Portal	Cell+Portal
Directionality	Full	1st order	1st order	Full	1st order	1st order
Decomposition	Automatic	Automatic	Automatic	Manual	Manual	Manual

Figure 8: Method comparison: *Ours is the only methods that allows simulation of many sources on large scenes at realtime rates.*

Frequency and time decomposition: Other solutions introduced frequency-domain precomputation based on acoustic rendering equation [Siltanen et al. 2009] but were limited to static sources. Recently, [Antani et al. 2012] extended the approach by precomputing acoustic transfer operators. This approach can only handle a few moving sources, is limited to diffuse reflections and its storage overhead (50-100X) is quite high as compared to our method.

Numerical propagation: Numeric based precomputation approaches [Raghuvanshi et al. 2010] are more accurate than GA methods and can model high order diffraction and scattering effects. However, the complexity increases as a fourth power of frequency and a linear function of the volume. As a result, they are limited to small indoor scenes. The accuracy of our approach is ultimately limited by the use of GA for the simulation step. However, while it relies on geometrical criteria, we believe our similarity measure and adaptive sampling could be used independently from the actual simulation technique and provide speed-up to previous approaches such as [Raghuvanshi et al. 2010].

Cell and portals: Many games and interactive applications use cell-and-portal scene decompositions and this can be utilized to precompute higher-order reflections of sound between moving sources and listeners using ray tracing [Foale and Vamplew 2007; Stavrakis et al. 2008; Tsingos 2009]. These approaches typically store IRs sampled at a single position for each cell and/or portal encountered along the paths between the source and the listener but require significant manual intervention to define regions and portals which is unpractical for large-scale environments. Defining cells and portal suitable for acoustic rendering can often be unintuitive, in particular for outdoor situations. In contrast, our solution provides an implicit and automatic definition of regions without manual intervention.

6.2 Limitations

Our approach introduces several approximations to the reverberation process. First, we render integrated decay profiles reconstructed with random phase, instead of the original impulse responses. This reduces the accuracy with which the early reflections can be rendered. In particular, flutter echoes might not be captured by this approach unless the number of sub-bands is increased. However, it is sufficient to capture relevant acoustical cues such as surface proximity effects and diffraction by obstacles. Second, the late reverberation is rendered on the client using an approximation to the dry cluster signal. This was found to have very limited impact on the perceived quality of the reverberation as the reverberation filter itself is modeled with random-phase. In the supplemental material, we provide several examples comparing the result of the convolution with a reference binaural impulse response generated from our simulation to our deferred reverberation processing using the corresponding decay profile.

Since are similarity reduction process is a heuristic based on the geometric data that forms the first order response, it may fail in some case to accurately estimate the late response. In scenes where the depth variance is very large, the early response time cannot be reliably estimated from local geometry. Also, in scenes with large amounts of occlusion, the earliest response is often the result of diffraction paths and our method may not handle these scenes robustly.

Additionally, our reduction metrics will be overly conservative in some scenes. In scenes with small enclosed areas, our method may not find similar points even if the regions are acoustically similar, since line-of-sight is used to determine sample point merging. Cases which cause the similarity heuristic to fail can result in under sampling and over sampling, possibly resulting in environment acoustic maps that are not accurate or larger than necessary.

7 Conclusion and Future work

We have presented a VoIP system that can scale to thousands of clients in massive environments. We provide early and late acoustic responses appropriate to the environment complete with diffraction and reverberation effects. With an efficient server-side GPU implementation, our system can perform well with low hardware cost.

In the future, we plan on investigating additions to adjust the acoustic response based on dynamic objects using precomputed filters. We also would like to study possible perceptual reduction methods to further reduce the number of samples that must be stored.

References

ANTANI, L., CHANDAK, A., SAVIOJA, L., AND MANOCHA, D. 2012. Interactive sound propagation using compact acoustic transfer operators. *ACM Trans. Graph. 31*, 1 (Feb.), 7:1–7:12.

BAILEY, R., AND BRUMITT, B. 2010. Method and system for automatically generating world environment reverberation from a game geometry. Tech. rep., U.S. Patent Application US 2010/0008513 A1, January.

BEGAULT, D. R. 1994. *3D Sound for Virtual Reality and Multimedia*. Academic Press Professional.

BENESTY, J., GAENSLER, T., AND ENEROTH, P. 2000. Multi-channel sound, acoustic echo cancellation, and multi-channel

time-domain adaptive filtering. In *Acoustic Signal Processing for Telecommunication*. Kluwer Academic Publishers, 101–120.

BOUSTEAD, P., AND SAFAEI, F. 2004. Comparison of delivery architectures for immersive audio in crowded networked games. In *Proceedings of the 14th international workshop on Network and operating systems support for digital audio and video*, ACM, New York, NY, USA, NOSSDAV '04, 22–27.

CHANDAK, A., ANTANI, L., TAYLOR, M., AND MANOCHA, D. 2009. Fastv: From-point visibility culling on complex models. In *Eurographics Symposium on Rendering*.

FOALE, C., AND VAMPLEW, P. 2007. Portal-based sound propagation for first-person computer games. In *Proceedings of the 4th Australasian conference on Interactive entertainment*, IE '07, 9:1–9:8.

FUNKHOUSER, T., CARLBOM, I., ELKO, G., PINGALI, G., SONDHI, M., AND WEST, J. 1998. A beam tracing approach to acoustic modeling for interactive virtual environments. In *Proc. of ACM SIGGRAPH*, 21–32.

FUNKHOUSER, T. A., MIN, P., AND CARLBOM, I. 1999. Real-time acoustic modeling for distributed virtual environments. In *Proc. of ACM SIGGRAPH*, 365–374.

GOODE, B. 2002. Voice over internet protocol (voip). In *Proceedings of the IEEE*, 1495 – 1517.

GREENE, N. 1986. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications 6*, 11 (Nov.).

GREENHALGH, C., AND BENFORD, S. 1995. Massive: a collaborative virtual environment for teleconferencing. *ACM Trans. Comput.-Hum. Interact. 2* (September), 239–261.

HALLORAN, J. 2009. It's talk, but not as we know it: Using voip to communicate in war games. In *Proceedings of the 2009 Conference in Games and Virtual Worlds for Serious Applications*, VS-GAMES '09, 133–140.

HOLLIER, M. P., RIMELL, A. N., AND BURRASTON, D. 1997. Spatial audio technology for telepresence. *BT Technology Journal 15*, 4, 33 – 41.

JOT, J.-M. 1999. Real-time spatial processing of sounds for music, multimedia and interactive human-computer interfaces. *Multimedia Systems 7*, 1, 55–69.

MARKOPOULOU, A., TOBAGI, F., AND KARAM, M. 2002. Assessment of voip quality over internet backbones. In *IEEE INFOCOM 2002*, 150 – 159.

MERIMAA, J., AND PULLKI, V. 2004. Spatial impulse response rendering. *Proc. of the 7th Intl. Conf. on Digital Audio Effects (DAFX'04)* (Oct.).

POPE, J., CREASEY, D., AND CHALMERS, A. 1999. Realtime room acoustics using ambisonics. *Proc. of the AES 16th Intl. Conf. on Spatial Sound Reproduction*, 427–435.

RADENKOVIC, M., GREENHALGH, C., AND BENFORD, S. 2002. Deployment issues for multi-user audio support in cves. In *Proceedings of the ACM symposium on Virtual reality software and technology*, ACM, New York, NY, USA, VRST '02, 179–185.

RAGHUVANSHI, N., SNYDER, J., MEHRA, R., LIN, M., AND GOVINDARAJU, N. 2010. Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes. In *ACM Trans. on Graphics*, vol. 29, 68:1 – 68:11.

SAFAEI, F. 2005. Dice: Internet delivery of immersive voice communication for crowded virtual spaces. In *Proceedings of the 2005 IEEE Conference 2005 on Virtual Reality*, IEEE Computer Society, Washington, DC, USA, VR '05, 35–41.

SALLNÄS, E.-L. 2005. Effects of communication mode on social presence, virtual presence, and performance in collaborative virtual environments. *Presence: Teleoper. Virtual Environ. 14*, 4, 434–449.

SILTANEN, S., LOKKI, T., AND SAVIOJA, L. 2009. Frequency domain acoustic radiance transfer for real-time auralization. *Acta Acustica 95*, 1, 106–117.

STAVRAKIS, E., TSINGOS, N., AND CALAMIA, P. 2008. Topological sound propagation with reverberation graphs. *Acta Acustica/Acustica - the Journal of the European Acoustics Association 94*, 921–932.

TAYLOR, M., CHANDAK, A., ANTANI, L., AND MANOCHA, D. 2009. Resound: interactive sound rendering for dynamic virtual environments. In *Proc. of the seventeen ACM international conference on Multimedia*, ACM, New York, NY, USA, 271–280.

TSINGOS, N., DACHSBACHER, C., LEFEBVRE, S., AND DELLEPIANE, M. 2007. Instant sound scattering. In *Proc. of the Eurographics Symposium on Rendering*, 111–120.

TSINGOS, N. 2009. Pre-computing geometry-based reverberation effects for games. *35th AES Conference on Audio for Games*.

VORLÄNDER, M. 1989. Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm. *The Journal of the Acoustical Society of America 86*, 1, 172–178.

WADLEY, G., GIBBS, M., AND BENDA, P. 2007. Speaking in character: using voice-over-ip to communicate within mmorpgs. In *Proceedings of the 4th Australasian conference on Interactive entertainment*, IE '07, 24:1–24:8.

WEST, J., BLAUERT, J., AND MACLEAN, D. 1992. Teleconferencing system using head-related signals. *Applied Acoustics 36*, 3-4, 327 – 333.

WILLIAMS, D., CAPLAN, S., AND XIONG, L. 2007. Can You Hear Me Now? The Impact of Voice in an Online Gaming Community. *Human Communication Research*, 4, 427–449.