Fast Geometric Sound Propagation with Finite-Edge Diffraction

Lakulish Antani* Anish Chandak† Micah Taylor‡ Dinesh Manocha§

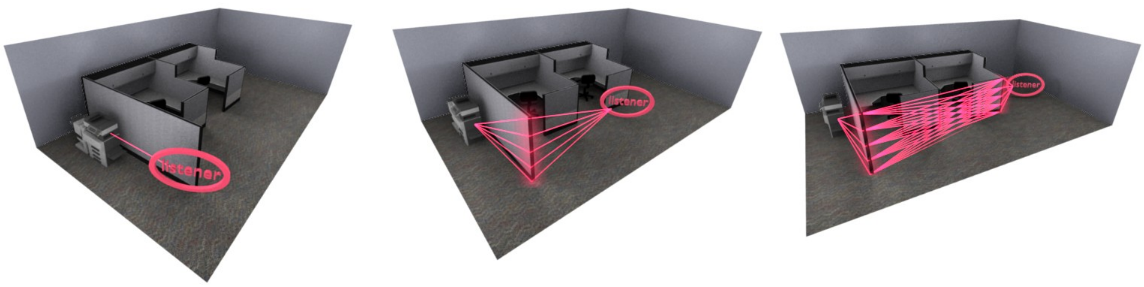


Figure 1: *Propagation paths for the sound of an office printer as it diffracts around cubicle edges and reaches the listener. Left to right: (a) direct sound (b) first-order diffraction (c) second-order diffraction.*

Abstract

We present a fast algorithm to perform sound propagation in complex 3D scenes. Our approach computes propagation paths from each source to the listener by taking into account specular reflections and higher-order edge diffractions around finite edges in the scene. We use the well known Biot-Tolstoy-Medwin diffraction model along with efficient algorithms for region-based visibility to cull away primitives and significantly reduce the number of edge pairs that need to be processed. The performance of region-based visibility computation is improved by using a fast occluder selection algorithm that can combine small, connected triangles to form large occluders and perform conservative computations at object-space precision. We show that our approach is able to reduce the number of visible primitives considered for sound propagation by a factor of 2 to 4 for second order edge diffraction as compared to prior propagation algorithms. We demonstrate and analyze its performance on multiple benchmarks.

Keywords: visibility, object-space, from-region, sound propagation, diffraction

1 Introduction

Sound rendering or auditory displays can augment graphical rendering and provide the user with an enhanced spatial sense of presence. Some of the driving applications of sound rendering include acoustic design of architectural models or outdoor scenes, walkthroughs of large CAD models with sounds of machine parts or moving people, urban scenes with traffic, training systems, computer games, etc. A key component in these applications is accurate computation of sound propagation paths, which takes into account the knowledge of sound sources, listener locations, the 3D model of the environment, and material absorption and scattering properties.

There is extensive literature on simulating the propagation of sound, including reflections and diffraction. The propagation of sound in a medium is governed by the *acoustic wave equation*, a second-order partial differential equation [Svensson and Kristiansen 2002]. However, numerical methods that directly solve the acoustic wave equation can take tens of minutes even for simple rooms. On the other hand, fast sound propagation methods use geometric techniques such as ray tracing or volumetric tracing which work well in terms of handling specular reflections, and can take advantage of recent advances in real-time ray tracing techniques.

However, current methods are either not fast enough for interactive applications or may not compute all propagation paths accurately. As a result, interactive applications such as computer games tend to use statically designed environment reverberation filters that are computed based on occlusion and obstruction between the sound source and the listener. Some games use the notion of audio shaders that identify the surrounding geometric primitives and dynamically adjust the time delays of the direct sound and the audio responses received from sound reflections.

In this paper, we primarily focus on simulating edge diffraction for geometric sound propagation. Diffraction is an important effect that causes sound to scatter when encountering the finite boundaries of relatively large objects, resulting in audio energy being propagated to positions that are out of line-of-sight from the source. Diffraction effects also affect the sound field at positions in line-of-sight from the source. In acoustic simulation, diffraction effects are primarily modeled at the edges of the objects in the scene. The computation of diffraction effects can convey important audio cues from sources that are not visible to the listener, and allow more listener positions to receive contributions from the sound source. It is necessary to simulate diffraction accurately in order to obtain a more realistic and smooth transition, especially when the listener or the source is moving.

In the context of geometric propagation, two main approaches exist for modeling edge diffraction using geometric acoustics techniques: the Uniform Theory of Diffraction (UTD) [Kouyoumjian and Pathak 1974] and the Biot-Tolstoy-Medwin (BTM) [Biot and Tolstoy 1957; Medwin et al. 1982] model. UTD models diffraction around an infinite edge in terms of a single virtual point source. While this makes it fast enough to be useful in interactive applications [Tsingos et al. 2001; Taylor et al. 2009], it is an approximate method and may only work well for large models in outdoor scenes.

*lakulish@cs.unc.edu
†achandak@cs.unc.edu
‡taylormt@cs.unc.edu
§dm@cs.unc.edu

On the other hand, BTM models diffraction around *finite* edges in terms of many virtual point sources located along the edge. This makes it more accurate than UTD, but also more computationally intensive.

Main Results We present an algorithm for fast geometric sound propagation based on the BTM model in static scenes with moving sources and listener positions. Our approach is based on the fact that a BTM-based propagation algorithm requires the capability to determine which other diffracting edges are visible from a given diffracting edge. This reduces to a *from-region visibility* problem, and we use a conservative from-region visibility algorithm which can compute the set of visible triangles and edges at object-space precision in a conservative manner. We also present a novel occluder selection algorithm that can improve the performance of from-region visibility computation.

The main contributions of this paper are as follows:

- **Accelerated higher-order BTM diffraction.** We present a fast algorithm to accurately compute the first few orders of diffraction using the BTM model. We use object-space conservative from-region visibility to significantly reduce the number of edge pairs that need to be considered as compared to the state-of-the-art for second order diffraction. We demonstrate that for scenes of complexities typically encountered in interactive sound propagation applications, our approach can use visibility information to reduce this number by a factor of 2 to 4.
- **Effective occluder selection for region-based visibility.** We present a fast algorithm for occluder selection that can compute occluders in all directions around a given convex region. Our algorithm can combine small, connected sets of primitives into large occluders. The final set of visible primitives is then computed using state-of-the-art occlusion culling techniques. We demonstrate that our occluder selection technique is able to quickly generate occluders consisting of 2-6 triangles each on the average in complex scenes in a few seconds per visibility query on a single core.

We show that our approach is able to reduce the amount of visible geometry considered by sound propagation algorithms by a factor of 2 to 4 for second order edge diffraction. This allows us to obtain a speedup factor of 2 to 4 when simulating second edge diffraction using the BTM model on our benchmark scenes.

Outline The rest of this paper is organized as follows: Section 2 describes background material and related work. Section 3 presents our diffraction algorithm and how we use visibility computations to accelerate the diffraction computations. Section 4 describes our novel occluder selection technique and its use to improve the performance of from-region visibility algorithms. Section 5 describes our implementation and presents experimental results.

2 Background

In this section, we give a brief overview of background material on sound propagation algorithms, region-based visibility computation and image-source methods for geometric sound propagation.

2.1 Sound Propagation

The acoustic properties of a scene are described using the *impulse response* (IR). The IR is computed at the listener's position, and represents the pressure signal arriving at the listener for a unit impulse signal emitted by the isotropic point source. The IR is a linear transform, which implies that given an arbitrary anechoic sound signal emitted by the source, the signal received by the listener (taking into account propagation effects) can be obtained by convolving the anechoic signal with the impulse response.

The propagation of sound in a medium is governed by the *acoustic wave equation*, a second-order partial differential equation [Svensson and Kristiansen 2002]. Several methods exist that directly solve the wave equation using numerical methods [Ciskowski and Brebbia 1991; Lehtinen 2003] and accurately model sound propagation in a scene. However, despite recent advances [Raghuvanshi et al. 2008], these methods can take many minutes to compute the impulse responses and can be too slow for practical applications.

Most sound propagation techniques used in practical applications model the acoustic effects of an environment using linearly propagating rays. These *geometric acoustics* (GA) techniques are not as accurate as numerical methods in terms of solving the wave equation, and cannot easily model all kinds of propagation effects, but they allow simulation of early reflections at real-time rates.

Specular reflections are easy to model using GA methods. The most common methods include the image source method [Allen and Berkley 1979; Funkhouser et al. 1998; Schröder and Lentz 2006; Laine et al. 2009], ray tracing [Krokstad et al. 1968; Vorlander 1989] and approximate volume tracing [Lauterbach et al. 2007; Chandak et al. 2008]. Of these methods, the image source method is the most accurate, since (if implemented correctly) it is guaranteed to not miss any specular propagation paths between source and listener. GA methods also exist for modeling diffuse reflections. The two main techniques of doing so are based on path tracing [Dalenbäck 1996; Kapralos et al. 2004] and radiosity [Siltanen et al. 2007; Siltanen et al. 2009].

Diffraction is relatively difficult to model using GA techniques (as compared to specular reflections), because it involves sound waves bending around objects. The two commonly used geometric models of diffraction are the Uniform Theory of Diffraction (UTD) [Kouyoumjian and Pathak 1974] and the Biot-Tolstoy-Medwin (BTM) model [Svensson et al. 1999]. The UTD model assumes infinite diffracting edges, an assumption which may not be applicable in real-world scenes (e.g., indoor scenes). However, UTD has been used successfully in interactive applications [Tsingos et al. 2001; Antonacci et al. 2004; Taylor et al. 2009]. BTM, on the other hand, deals with finite diffracting edges, and therefore is more accurate than UTD; however it is much more complicated and has only recently been used – with several approximations – in interactive applications [Schröder and Pohl 2009].

2.2 Image Source Method for Geometric Propagation

Given a point source S and a listener L, it is easy to check if a direct path exists from S to L. This is a ray shooting problem. The basic idea behind the image source method is as follows. For a specular reflector (in our case, a triangle) T, a specular path $S \rightarrow T \rightarrow L$ exists if and only if a direct path exists from the *image* of S formed by T, to L. In the absence of any visibility information, image sources need to be computed about *every* triangle in the scene. This process can be applied recursively to check for higher order specular paths from S to L, but the complexity can increase exponentially as a function of the number of reflections.

For a given source position, this process can be accelerated [Laine et al. 2009] as follows. Note that first-order image sources only need to be computed about triangles visible to S. For a first-order image source S_1, second-order image sources only need to be computed for the triangles that are visible to S_1 through T, and so on for higher order image sources. It is also possible to integrate geometric models for edge diffraction into the image source framework [Pulkki et al. 2002]. In Section 3 we describe our method that uses from-point and from-region visibility algorithms to accelerate the GA algorithms which integrate edge diffraction effects into the image source method.

2.3 From-Region Visibility

Visibility computation is one of the classic problems studied extensively due to its importance in many fields such as computer graphics, computational geometry, and robotics. The problem of finding surfaces visible from a given *region*, such as a triangle, edge, or bounding box (i.e., the from-region visibility problem) is well-studied in the literature. Exact solutions can be computed using techniques such as aspect graphs [Gigus et al. 1991], visibility complex [Durand et al. 1996; Durand et al. 1997] or by computing unobstructed rays by performing CSG operations in a dual line space [Nirenstein et al. 2002]. These methods have high complexity – $O(n^9)$ for aspect graphs and $O(n^4)$ for the visibility complex, where n is the number of scene primitives – and are too slow to be of practical use on complex models.

Many methods exist to compute approximate visibility by essentially sampling the space of rays originating in the query region. These methods are fast enough to be practically useful on large and complex models [Wonka et al. 2006; Bittner et al. 2009], but have one important limitation: they compute a *subset* of the exact solution (i.e., approximate visibility), and therefore, are limited to sampling-based applications such as interactive graphical rendering, and may not provide sufficient accuracy for sound rendering. This is because the image source method requires us to find all possible propagation paths (see Section 2.2), which in turn requires the visibility algorithm to not miss any visible geometry. For a sampling-based algorithm in complex scenes, this can require a prohibitively high sampling frequency in order to guarantee that all visible geometry is returned in the output of the algorithm.

The other class of applicable algorithms is *conservative* visibility algorithms. These algorithms can efficiently compute a *superset* of the exact solution. Conservative algorithms operate in dual ray space by finding *stabbing lines* [Teller and Séquin 1991] or in primal space by performing *occlusion culling* with respect to *shadow frusta* [Durand et al. 2000; Chhugani et al. 2005].

3 Sound Propagation

Our geometric sound propagation algorithm is based on the image source method [Allen and Berkley 1979; Schröder and Lentz 2006]. As originally formulated, this technique can simulate specular reflections only. However, it is possible to extend this method to handle edge diffraction effects as well [Pulkki et al. 2002; Calamia et al. 2005]. In this section, we present our efficient algorithm to perform diffraction using region-based visibility computations to accelerate the computations. Figure 2 gives an overview of our technique.

3.1 Edge Diffraction and Image Sources

We now briefly outline a method of integrating edge diffraction modeling into the image source method [Pulkki et al. 2002]. Analogous to how specular reflection about a triangle is modelled by computing the image of the source with respect to the triangle, diffraction about an edge is modelled by computing the image of the source *with respect to the edge*. In the rest of the paper, we use

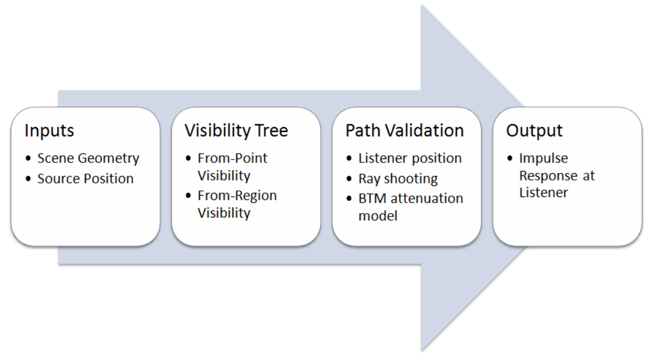


Figure 2: *Overview of our sound propagation algorithm. Using the scene geometry and source position as input, we first construct a visibility tree describing potential propagation paths. Next, we use the listener position to find valid propagation paths using the visibility tree. Finally, we use the valid paths and the BTM model to compute the impulse response at the listener.*

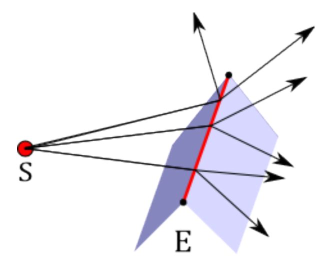


Figure 3: *Image source of a diffracting edge. Sound from source S scatters in all directions upon encountering diffracting edge E. E itself is therefore the image source of S about E. The fact that rays scatter in all directions from E implies that from-region visibility is required to compute all geometry reachable by these rays.*

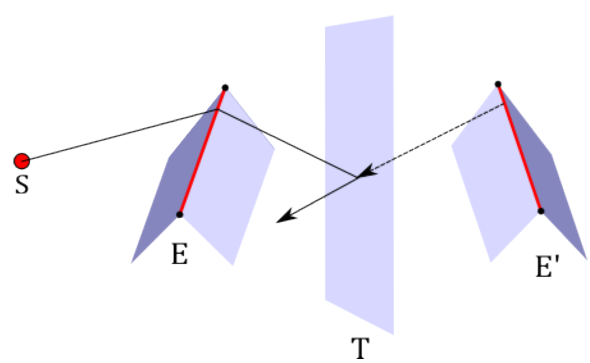


Figure 4: *Image sources for one diffraction followed by one specular reflection. S is the source and E is a diffracting edge. T is a specular reflector. E induces a diffraction image source along its length. This is reflected in the plane of T to give E', which lies along the reflection of E in T.*

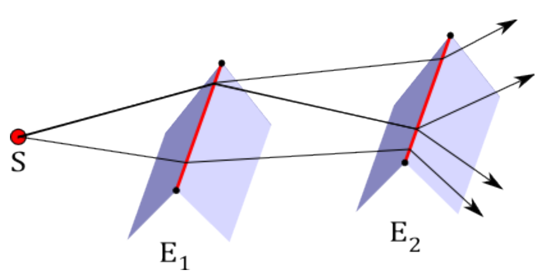


Figure 5: *Image sources for two successive diffractions. S is the source and E_1 and E_2 are diffracting edges. E_1 induces a first-order diffraction image source along its length. E_2 induces a second-order image source along its length.*

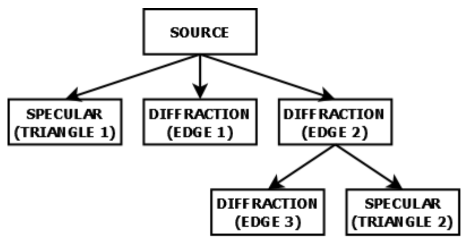


Figure 6: *Visibility tree. Each node is labelled with a triangle (for specular reflections) or an edge (for edge diffractions). Each path in this tree corresponds to a sequence of triangles and/or edges that can be encountered by a ray propagating from source to listener.*

the term "source" to refer to actual sound sources in the scene as well as image sources of any order.

The key idea is that the image source of a point source S with respect to diffracting edge E is that edge E itself (see Figure 3). This is based on the Huygens interpretation of diffraction [Medwin et al. 1982]. (Intuitively, one can think of modelling the diffraction about the edge in terms of infinitesimally small emitters located along the edge.) This means that image sources can now be points or line segments. It follows from the Huygens interpretation that the image of a line source E_1 about a diffracting edge E_2 is E_2 (see Figure 5). Further note that the image of a point or line source S_i about a planar specular reflector T is obtained by reflecting S_i across the plane of T (see Figure 4).

3.2 Visibility and Image Sources

Note that we only need to compute image sources for a source S_i about triangles and/or edges that are visible to S_i. If S_i is a point source, this involves from-point visibility computation and conservative computation of the visible primitives at object-precision. If S_i is a line or edge source, however, we require from-region visibility computation, specifically, from-edge visibility computation. This visibility computation makes the BTM model much more complicated than the UTD model. In order to not miss potential propagation paths when computing image sources, we require object-precision from-region visibility, for which exact algorithms are complicated and slow. If we use image-space visibility algorithms they can either miss propagation paths or result in aliasing artifacts.

In practice, most existing BTM implementations either approximate visibility information, or use overly conservative culling techniques. For example, the MATLAB Edge Diffraction toolbox is the state-of-the-art BTM implementation [Svensson 1999]. For any edge E formed by planes P_1 and P_2, the toolbox implementation culls away edges whose both endpoints are behind both P_1 and P_2. This is analogous to view frustum culling in graphics. In contrast, our approach uses a conservative from-region visibility algorithm to perform occlusion culling, so as to cull away additional geometry that is known to be invisible from E.

We use a two-step approach based on the image source method [Laine et al. 2009]. First, for a given source position S, we construct a *visibility tree* $VT(S,k)$ upto a user-specified depth k (see Figure 6). Each path in $VT(S,k)$ is a sequence of (upto k) triangles and/or edges that a ray starting from S reflects and/or diffracts about as it reaches the listener at any position L. In other words, the paths in $VT(S,k)$ *partition* the set of propagation paths from S to L, with each path P_t in $VT(S,k)$ corresponding to an equivalence class $R(P_t)$ of propagation paths between S and L. Next, given a listener position L, we traverse the visibility tree, and for each path P_t in $VT(S,k)$, we determine which of the propagation paths in $R(P_t)$ are valid (i.e., unoccluded by other primitives) for the given source/listener pair. We refer to the second step of the process as *path validation*.

Each node in the tree corresponds to an image source S_i. Denote the node corresponding to S_i by $N(S_i)$. We begin by creating a single node $N(S)$ corresponding to the source position S. The tree is then built recursively. To compute the children of $N(S_i)$, we compute the set of triangles $T(S_i)$ and edges $E(S_i)$ visible from S_i. For each $t \in T(S_i)$ we reflect S_i about t, obtaining the reflection image source S_i^t, and construct the child node $N(S_i^t)$. For each $e \in E(S_i)$, we construct the child node $N(e)$. Note that computing $T(S_i)$ and $E(S_i)$ requires a from-point visibility query from S_i if it is a point source, or a from-region visibility query if it is a line or edge source. We stop construction of the tree beyond a given maximum depth. This maximum depth can be user specified in our implementation.

Note that the visibility tree essentially describes the search space of propagation paths that need to be considered when computing the impulse response at L. To ensure that we consider all possible diffraction paths between S and L, we need to ensure that we do not miss any of the visible edges when constructing the visibility tree. One way to ensure this is to assume each edge is visible from every other edge, or use the simple plane culling approach used by the MATLAB toolbox. However, this means that each node $N(S_i)$ corresponding to an edge source S_i will have a very large number of children, many of which may not be reachable by a ray starting on S_i. This can dramatically increase the branching factor of the nodes in the tree, making higher-order paths almost impractical to compute. Therefore, we require conservative visibility algorithms for both from-point and from-region queries that are not overly conservative.

Another important point to note is that the tree must be reconstructed if the source moves. However, if the scene is static, all necessary from-region visibility information can be precomputed, allowing the tree to be rebuilt quickly. In the next section, we briefly describe the path validation process required to apply the BTM model for edge diffraction.

3.3 Path Validation

After constructing the visibility tree for a given source position, the next step is to use the tree to find propagation paths between the source and the listener, and to compute contributions from these paths to the final impulse response at the listener position. We use the model described by [Svensson et al. 1999], where the impulse response given a source at S and listener L and a single diffracting wedge is given by:

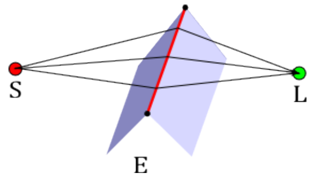


Figure 7: *Diffraction paths between source S and listener L across edge E. Note that here $n = 3$ ray shooting tests are needed to validate the diffraction paths.*

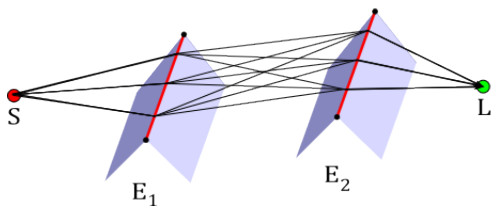


Figure 8: *Second order diffraction paths between source S and listener L across edges E_1 and E_2. Note that here $n = 3$ ray shooting tests are needed between S and E_1 and between E_2 and L, whereas $n^2 = 9$ tests are required between E_1 and E_2.*

$$h(t) = -\frac{v}{4\pi} \int_{z_1}^{z_2} \delta \left(t - \frac{m(z) + l(z)}{c} \right) \frac{\beta(S, z, L)}{m(z)l(z)} dz \quad (1)$$

where v is the *wedge index* for the diffracting edge [Svensson et al. 1999], z_1 and z_2 are the endpoints of the edge, z is a point on the edge, $m(z)$ is the distance between S and z, $l(z)$ is the distance between z and R and $\beta(S, z, L)$ is essentially the diffraction attenuation along a path from S to z to L. We evaluate this integral by discretizing the edge into some n pieces and assuming that the integrand has a constant value over each piece (equal to its value at the midpoint of the piece). For each edge piece this gives an attenuation of:

$$h_i = -\frac{v}{4\pi} \frac{V(S, z_i)V(z_i, L)\beta(S, z_i, L)}{m(z_i)l(z_i)} \Delta z_i \quad (2)$$

where h_i is the IR contribution caused by edge sample i with midpoint z_i, and $V(x, y)$ is a Boolean valued visibility function which is true iff the ray from point y to point x is unoccluded by scene geometry. For second order diffraction, the corresponding attenuation is:

$$\begin{aligned} h_{ij} = & \frac{v_1 v_2}{16\pi^2} V(S, z_i)V(z_i, z_j)V(z_j, L) \\ & \times \frac{\beta(S, z_i, z_j)\beta(z_i, z_j, L)}{m_1(z_i)m_2(z_i, z_j)l(z_j)} \Delta z_i \Delta z_j \end{aligned} \quad (3)$$

where h_{ij} is the IR contribution from sample i on the first edge and sample j on the second edge, with midpoints z_i and z_j respectively. Here, v_1 is the wedge index for the first edge and v_2 is the wedge index for the second edge. Given a path in the visibility tree which may contain any number of specular and/or diffraction nodes, we wish to use the Equations 2 and 3 to compute contributions to the final IR. For a given listener position L, we perform this step as follows:

1. We traverse each path in the tree in a bottom-up manner.

2. For each leaf node $N(S_l)$, we compute all valid propagation paths between S_l and L. For each internal node $N(S_i)$ and its parent $N(S_j)$, we compute all valid propagation path segments between S_i and S_j.

3. For each valid path segment with endpoints p_i and p_j, we compute the corresponding delay $\|p_j - p_i\| / c$, distance attenuation $1/\|p_j - p_i\|$, specular attenuation α where α is the specular coefficient of the reflecting triangle (if S_j is a specular node) and diffraction attenuation $\beta(p_i, p_j, p_k)$ where p_k is the path endpoint corresponding to the parent of S_j (if S_j is a diffraction node).

In practice, these delays and attenuations are computed only if the corresponding visibility terms are nonzero. This check is performed using ray shooting between S_i and S_j. Ideally, we would like to compute the set of all unoccluded rays between S_i and S_j. (If S_j is formed by a specular reflector T, then we only consider the rays between S_i and S_j which intersect T and are unoccluded between S_i and their hit point on T.) If S_i and S_j are both point sources, this reduces to a simple ray shooting test. However, if either one is a line source, path validation reduces to from-region visibility computation.

In order to compute accurate contributions from each propagation path, we would ideally need to compute exact visibility information. However, note that the BTM model computes the effect of diffraction about an edge in terms of a line integral over the edge. This integral must be discretized in order to compute impulse responses. We approximate the line integral using the midpoint method – by dividing the edge into n segments, and computing contributions due to paths passing through the midpoints of each segment. This method of integration allows us to use n ray shooting tests (one for the midpoint of each of the n edge segments) to compute (approximate) visibility.

Observe that a propagation path is essentially a polyline which starts at the source, ends at the listener and whose intermediate vertices lie on triangles and/or edges in the scene. In the case of specular reflections only, path validation is performed using ray shooting to validate each segment of a polyline through the scene. If we also include one edge diffraction in the propagation path, we now need to validate n polylines through the scene, using n ray shooting tests for each image source along a path in the visibility tree. If we include a second edge, we need to validate n^2 polylines, and so on. However, in this case, we do not need to perform n^2 ray shooting tests for every image source along the path: only for image sources between the two diffracting edges (see Figures 7 and 8 for details). This is because there are n polylines from the source to the first edge, and n polylines from the second edge to the listener. Therefore the n^2 polylines from the source to the listener share several common segments, which allows us to reduce the number of ray shooting tests required. By a similar argument, it can be shown that for third- and higher-order diffraction paths, the number of ray shooting tests required between any two image sources is at most $O(n^2)$, even though the total number of polylines is $O(n^d)$ (where d is the number of diffracting edges in the path).

Once the validation step is complete and all contributions to the IR at the listener position have been computed, the next step is to render the final audio. We simply convolve the input sound signal with the computed impulse response to generate the output audio for a given listener position. To generate smooth audio for a moving listener, we interpolate between impulse responses at successive listener positions.

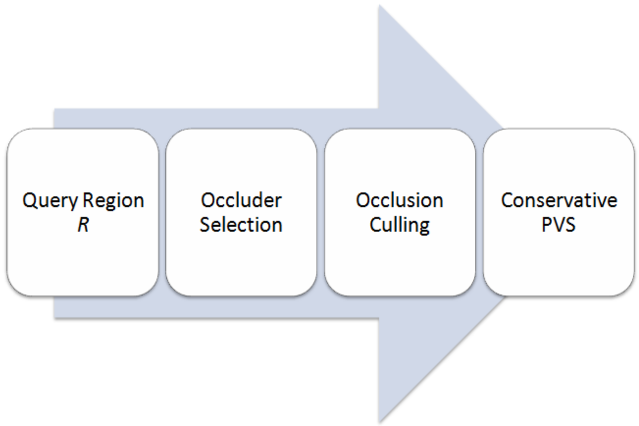


Figure 9: *Overview of our from-region visibility approach. In the first step, we choose occluders for the query region R. Next, we use the occluders to compute which primitives are hidden from R by the occluders. The set of primitives not hidden by the occluders is the potentially visible set for R.*

4 Visibility Computation

In this section, we present our region-based visibility computation algorithm that is used to speed up the edge diffraction computation (as highlighted in Section 3). Specifically, we present a novel algorithm to compute the occluders from a given region and combine it with prior methods to compute the potentially visible set (PVS) of primitives from a given region at object-space precision. Figure 9 shows an overview of our visibility algorithm.

Formally, the from-region visibility problem can be described as follows. Given a convex region $R \subset \mathbb{R}^3$ and a set of scene primitives Π, we wish to compute a subset of primitives $\pi \subseteq \Pi$ such that every primitive $p \in \Pi$ which is hit by a ray originating in R is included in π. π is called the *potentially visible set* (PVS) of R. The smallest such set is the *exact* PVS π_{exact} of R. Our algorithm returns a *conservative* PVS, i.e. a superset of the exact PVS ($\pi \supseteq \pi_{exact}$).

Our visibility technique can be divided into two steps: *occluder selection* for choosing primitives to be used as occluders for a given region R, and *occlusion culling* for computing the PVS of R given the set of occluders. Note that our algorithm is general and can be used to compute the PVS of any convex region, including line segments (edges), triangles and volumetric cells such as bounding boxes. For our purposes however, we only use the algorithm to compute the PVS of diffracting edges.

4.1 Occluder Selection

The first step in computing the PVS of convex region R is to compute the potential occluders for R. One option would be to simply use every primitive in the scene as an occluder, and use an occlusion culling algorithm that handles occluder fusion. In an ideal scenario, such an approach would result in a PVS that is as close as possible to π_{exact}. However, the main issue with such an approach, which limits its practical application, is that the cost of occlusion culling is typically a function of the number of occluders [Chhugani et al. 2005]. Most prior work on occluder selection uses heuristics based on distance, solid angles, or area of primitives [Coorg and Teller 1997; Hudson et al. 1997; Durand et al. 2000; Koltun and Cohen-Or 2000]. Although the methods compute a subset of Π for use as occluders, they are unable to exploit the connectivity information of primitives to find any arbitrary set of connected triangles as occluders.

Thus, we propose a novel from-region occluder selection algorithm which exploits the connectivity information between scene primitives whenever feasible. Our approach is general and applicable to all kinds of models including "polygon soup" models. We make no assumptions about the model or the connectivity of the polygons. (In our implementation, the models are assumed to be triangulated, however, this is not a restriction imposed by our algorithm.) If the model connectivity information is given or can be extracted, our algorithm can exploit that information to compute large occluders formed by connected sets of primitives for occluder selection.

Our technique can be viewed as a generalization of the conservative from-point visibility technique used in the FastV algorithm [Chandak et al. 2009]. FastV computes from-point visibility by constructing a cubical box around the query point R, subdividing each of its faces into multiple quad patches Q (where the number of quad patches can be user-specified), and constructing frusta $F(R, q)$ from each quad patch $q \in Q$ and R (see Figure 10). Each of these frusta is used to determine which portions of the scene are visible from the query point that use the relevant patch as the viewport. Formally, for each $q \in Q$ we wish to determine the set of primitives $p \in \Pi$ such that there exists a ray from R to some point on p which passes through q.

Given a frustum $f = F(R, q)$ (defined by its corner rays), the FastV algorithm tries to compute a *blocker* for f. In the context of FastV, a blocker is defined as a connected set of triangles such that any convex combination of the corner rays of f intersects some triangle in the blocker. FastV traverses the scene hierarchy, and whenever a triangle T is found that intersects f, it uses the connectivity information associated with T to compute if some set of triangles connected to T can also be used as a blocker for f. It is possible that there may be no such triangles. Therefore, once the traversal is completed, FastV returns at most one blocker for f and zero or more connected sets of triangles in front of the blocker which do not completely block f.

Consider generalizing the frustum construction approach of FastV to the from-region case (i.e., now R can be any convex region). We compute an oriented bounding box that encloses R and subdivide its faces into a user-specified number of quad patches Q. The next step is to determine primitives visible from R through each quad patch $q \in Q$ (see Figure 11). Formally, we wish to determine the set of primitives p such that there exists at least one ray from some point $r \in R$ to p which passes through q. Put another way, we wish to determine all points from which R is partially visible through q. This corresponds to the region in front of q and bounded by the set S of *separating planes* constructed between R and q [Coorg and Teller 1997] (see Figure 11).

Note that we orient the separating planes such that Q lies in the positive half-space (interior) defined by each separating plane $s \in S$. We then construct a *separating frustum* $f = F(R, q)$ bounded by S. We could use view frustum culling techniques to cull Π to f to estimate the PVS of R. However, this approach may compute a PVS π such that there exist primitives $p_1, p_2 \in \pi$ where p_1 occludes p_2 from R, and the resulting PVS would be too conservative. Instead, we use FastV to trace f (see Figure 11). (Note that if R is in fact a single point, our occluder selection algorithm reduces to FastV.) Ideally, we would like to trace all that rays that start on R and pass through q, and the set of primitives reached would approach π_{exact}. However, tracing f using FastV computes a *subset* of triangles visible from R through Q (i.e., computes $\pi \subseteq \pi_{exact}$). Therefore, after occluder selection, we use a conservative occlusion culling algorithm to compute a superset of the exact PVS.

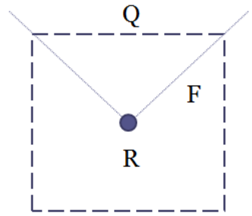


Figure 10: *Frustum construction performed by FastV. Given a query point R, we construct an axis-aligned box around it and divide the faces of the box into quad patches, one of which, Q, is shown in the figure. Given R and Q, we then trace a frustum F to compute the PVS for R.*

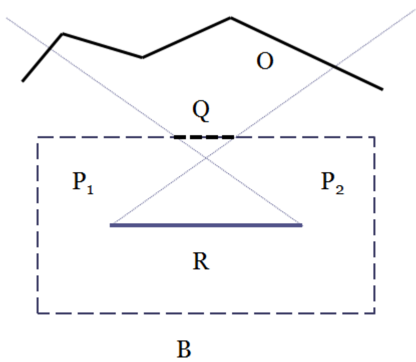


Figure 11: *Separating frustum construction, in 2D. Given a line segment R, we construct a fattened bounding box B, and divide its boundary into line segment patches, one of which is Q. We construct separating planes P_1 and P_2 between R and Q, and trace the frustum bounded by these planes and oriented such that Q is in the interior of the frustum. Here O is a blocker for the separating frustum, and is used as an occluder for R.*

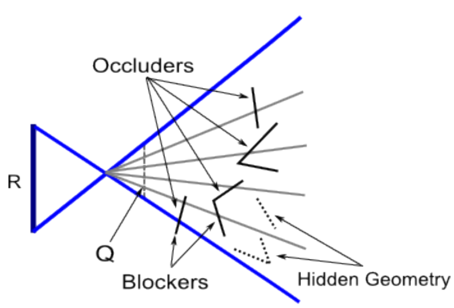


Figure 12: *Occluder selection in 2D. The bright blue lines are the corner rays of the separating frustum between query region R and quad patch Q. The grey lines indicate corner rays of sub-frusta formed by uniform frustum subdivision [Chandak et al. 2009]. Primitives chosen as occluders are shown as solid line segments, primitives hidden by the occluders are shown as dotted lien segments. Some of the occluders are frustum blockers, and these are also marked in the figure.*

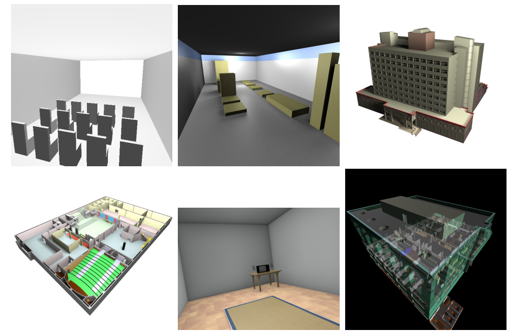


Figure 13: *Benchmarks. Clockwise from top left: (a) Room (876 triangles) (b) Factory (170 triangles) (c) Building (69K triangles) (d) Soda Hall (1.5M triangles) (e) House (1K triangles) (f) Floor (7.3K triangles).*

Tracing $f = F(R, q)$ using FastV can return a blocker for f. This blocker is a connected set of triangles such that any ray originating on R and passing through q intersects the blocker. Therefore, we use all blockers returned by FastV as occluders. However, it is possible that FastV may be unable to find a blocker for f. In such a case, we use the connected sets of triangles computed by FastV during scene traversal as occluders (see Figure 12 for an example).

4.2 PVS Computation

Given a set of occluders for R, the next step is to perform occlusion culling to compute the PVS of R. Ideally, we would like to determine the umbra of an occluder O with respect to a R. Unfortunately, the boundary of an exact umbra is bounded by curved surfaces [Teller 1992]. A common workaround is to compute a *shadow frustum* bounded by these curved surfaces, and use it to determine a subset of triangles occluded by O (thus computing a superset of the exact PVS for R). The shadow frustum is bounded by the *supporting planes* between R and O [Chhugani et al. 2005], and can be easily computed.

We can use any existing object-precision technique for occlusion culling, as long as it guarantees that the resulting PVS is conservative. Several methods that fit these requirements exist in the literature [Durand et al. 2000; Chhugani et al. 2005]. In our implementation, we have used a simple CPU-based frustum culling method. For each occluder O, we compute the shadow frustum $S(O, R)$ of O from R and mark all primitives behind O and completely contained in $S(O, R)$ as occluded from R. Once all shadow frusta have been processed in this manner, the primitives not marked hidden are added to the PVS of R.

5 Results

In this section, we present experimental results on sound propagation and from-region visibility. We compare our sound propagation system with the current state-of-the-art to highlight the benefits of using from-region visibility when computing sound propagation paths. Figure 13 shows the scenes we use to benchmark our code. The Room, Factory and House scenes are used to benchmark visibility tree construction. The Building, Floor and Soda Hall examples are complex scenes used to benchmark our occluder selection algorithm. Figure 17 shows some examples of diffraction paths computed by our algorithm. All of our tests were performed on high-end Intel Xeon workstations with 4GB RAM, running Windows Vista. Our implementation is written in C++ and uses SSE instructions to achieve high performance.

Visibility Tree Construction We first demonstrate the advantage of using from-region visibility in our BTM-based sound propagation system. We compare the performance of our visibility tree construction step (using from-region visibility) against visibility tree construction using only the simple culling approach used in the MATLAB Edge Diffraction toolbox [Svensson 1999] (as implemented in C++). We compare the time required to build the visibility tree as well as the size of the tree constructed for each approach. Our results are summarized in Table 1.

The table clearly highlights the importance of from-region occlusion culling in the BTM model. In the absence of occlusion culling, the size of the visibility grows very rapidly with depth. Our approach uses occlusion culling to essentially reduce the branching factor of the nodes of the visibility tree. Reducing the size of the tree in turn implies faster validation of diffraction paths using the BTM model.

Figure 14 shows the average percentage of total triangles (and diffracting edges) visible from the diffracting edges in various benchmark scenes. These plots clearly show that even in simple scenes which are typically used for interactive sound propagation, occlusion culling helps reduce the complexity of the visibility tree computed by our algorithm by a factor of 2 to 4.

Occluder Selection for From-Region Visibility We now turn to the performance of our from-region visibility implementation. Note that the following results are reported for from-triangle visibility. We report the running times of our occluder selection step per triangle in Table 2. These results were obtained for a single processor. The table also reports the average number of triangles in each occluder. This demonstrates how our occluder selection algorithm is able to effectively combine connected triangles into larger occluders. This results in larger occluders, which can potentially allow more triangles to be culled. Moreover, the computational cost of state-of-the-art from-region occlusion culling algorithms tends to increase with an increase in the number of occluders. For example, the vLOD system [Chhugani et al. 2005] constructs shadow frusta for each occluder and then solves for a single viewpoint contained in all of them. The time required for such computations can be reduced by using fewer, larger occluders formed by connected sets of triangles, such as those selected by our algorithm. This is because using fewer occluders implies that fewer shadow frusta need to be computed, and the viewpoint computation requires fewer frusta to be processed. Figure 16 shows the number of visible and culled triangles computed by our occlusion culling system for some triangles in two of our benchmark scenes.

Impulse Responses and Comparisons We have implemented the line integral formulation of the BTM model [Svensson et al. 1999] for performing path validation and computing impulse responses. The crucial parameter in the validation step is the number of samples each edge is divided into. A higher number of samples per edge results in more accurate evaluation of the BTM integral at a higher computational cost. Figure 15 shows impulse responses computed for diffraction about a simple double wedge for increasing numbers of samples per edge. As can be seen from the figure, increasing the number of samples causes the IRs to converge to the reference IR computed by the MATLAB toolbox [Svensson 1999] (also shown in Figure 15).

Further note that although the computational cost of the BTM model remains higher than that of the UTD model, it has been shown [Svensson et al. 1999] that the BTM model is more accurate than UTD model at low frequencies, where diffraction plays an important role. At low frequencies, numerical methods can be used to capture diffraction effects, but the complexity scales with the volume of the scene, as opposed to BTM-based methods whose complexity scales with the number of diffracting edges. Moreover, combining a numerical acoustics algorithm with geometric acoustics techniques for high frequency simulations remains a challenging problem, whereas the BTM approach can easily be combined with the image source method to compute accurate diffraction effects.

Scene	Triangles	Occluder Selection	
		Time (s)	Avg tris per occluder
Floor	7.3K	.12	6.0
Building	69K	1.3	3.0
Soda Hall	1.5M	14.8	6.7

Table 2: *Performance of our occluder selection algorithm for various benchmarks. All timings are reported for occluder selection for a single triangle (averaged over multiple triangles). The last column indicates the average size of occluders (in no. of triangles) returned by the occluder selection algorithm.*

Table 1 also shows the speedup obtained in the validation and IR computation step as a result of using conservative from-region visibility when constructing the visibility tree. As the table demonstrates, even for a very unoptimized implementation running on a single core, using conservative visibility algorithms can offer a significant performance advantage over state-of-the-art BTM-based edge diffraction modeling methods.

6 Conclusion

We have demonstrated the importance of conservative, object-space accurate from-region visibility in a geometric sound propagation system that can model specular reflections and edge diffractions. This is used to develop a fast sound propagation system. The approach is based on the image source method, and integrates edge diffraction into the image source framework. Edge diffractions are modeled using the Biot-Tolstoy-Medwin model. The set of potential propagation paths that need to be tested for validity is significantly reduced using fast conservative object-space from-region visibility techniques. This greatly accelerates the process of computing sound propagation paths and their contributions to the impulse response at the listener, leading to significant performance improvements over state-of-the-art geometric algorithms for modeling higher-order edge diffraction.

Our from-region visibility algorithm uses a novel, systematic occluder selection method that is fast and can assemble connected triangles into a single larger occluder. This allows for efficient occlusion culling using state-of-the-art techniques. Our approach is easy to parallelize and scales well on multi-core architectures. The modularity of our technique allows us to use our occluder selection algorithm with any from-region occlusion culling algorithm and gain the benefits of combining adjacent triangles into single occluders.

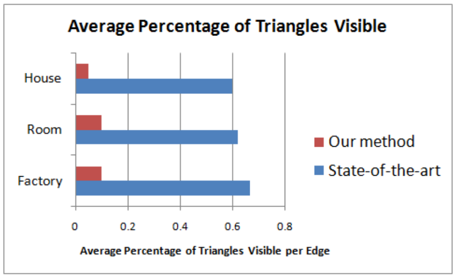
6.1 Limitations

Our approach has several limitations. It is possible that in the absence of large primitives that can be used as occluders, our algorithm would have to trace a large number of small frusta in order to select occluders, which could adversely affect its performance.

The BTM model is computationally intensive, and to the best of our knowledge, there exist no implementations of third- or higher-order edge diffraction based on it. However, there do exist special cases where it is necessary to model very high orders of edge diffraction;

Scene	Triangles	Edges	Second order diffraction paths in tree			BTM validation speedup	
			Our method	Toolbox	Size reduction	Edge sampling	Speedup
Factory	170	146	4424	12570	2.84	10×10	1.93
Room	876	652	43488	181314	4.17	5×5	3.23
House	1105	751	133751	393907	2.95	5×5	13.74

Table 1: *Advantage of using conservative from-region visibility for second order edge diffraction. Columns 4–6 demonstrate the benefit of using from-region visibility to cull away second order diffraction paths between mutually invisible edges. The last column shows the speedup caused by this reduction in the size of the visibility tree. Column 7 refers to the number of rays shot per edge and the number of integration intervals corresponding to each ray. For example, 5×5 refers to 5 rays shot per edge and a total of 25 integration intervals, with each ray shooting test used to compute the visibility term for 5 consecutive integration intervals.*



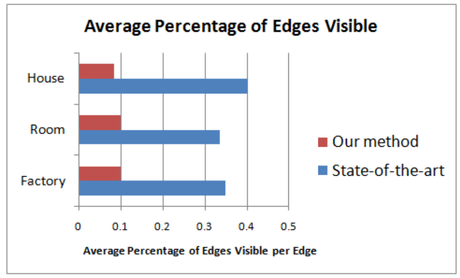


Figure 14: *Average amount of visible geometry returned by our approach as compared to the state-of-the-art for various benchmarks. The horizontal axis measures the fraction of visible geometry (triangles or edges, respectively) averaged over all edges in the scene. Smaller is better.*

one example is the case of sound diffracting around the highly tessellated surface of a large pillar.

6.2 Future Work

There are many possible avenues for future work. Our current implementation of our from-region visibility algorithm uses a simple object-space frustum culling technique for occlusion culling. This can cause it to miss cases of occluder fusion due to disconnected occluders. One possibility is to use conservative rasterization methods [Chhugani et al. 2005; Akenine-Moller and Aila 2005], which may be able to fuse such occluders and thereby result in a smaller PVS from a given region. Moreover, the occluder selection step itself can be implemented on the GPU for additional performance gains.

While our visibility tree construction step can construct paths of the form $source \to \cdots \to diffraction \to specular \to \cdots \to diffraction \cdots$, we discard such paths and do not compute IR contributions from them. Similarly, we discard paths with three or more edge diffractions. It would be a simple task to perform the visibility checks requires to compute which such paths are valid. However we are not aware of any BTM-based method for computing attenuations which can handle specular reflections between two edge diffractions, and therefore cannot compute contributions from such paths.

We use a simple midpoint method to evaluate the BTM integral and compute edge diffraction contributions to the final impulse response. However, the BTM integrand has poles which cannot be integrated across [Svensson and Calamia 2006; Calamia and Svensson 2007]. Our simple integration method does not account for these poles, and may integrate across them, leading to errors in the impulse response.

Acknowledgements

The authors would like to thank Paul Calamia and U. Peter Svensson for their invaluable feedback. This research is supported in part by ARO Contract W911NF-04-1-0088, NSF award 0636208, DARPA/RDECOM Contracts N61339-04-C-0043 and WR91CRB-08-C-0137, Intel, and Microsoft.

References

AKENINE-MOLLER, T., AND AILA, T. 2005. Conservative and tiled rasterization using a modified triangle setup. In *Journal of Graphics, GPU, and Game Tools*, vol. 10, 1–8.

ALLEN, J. B., AND BERKLEY, D. A. 1979. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America 65*, 4, 943–950.

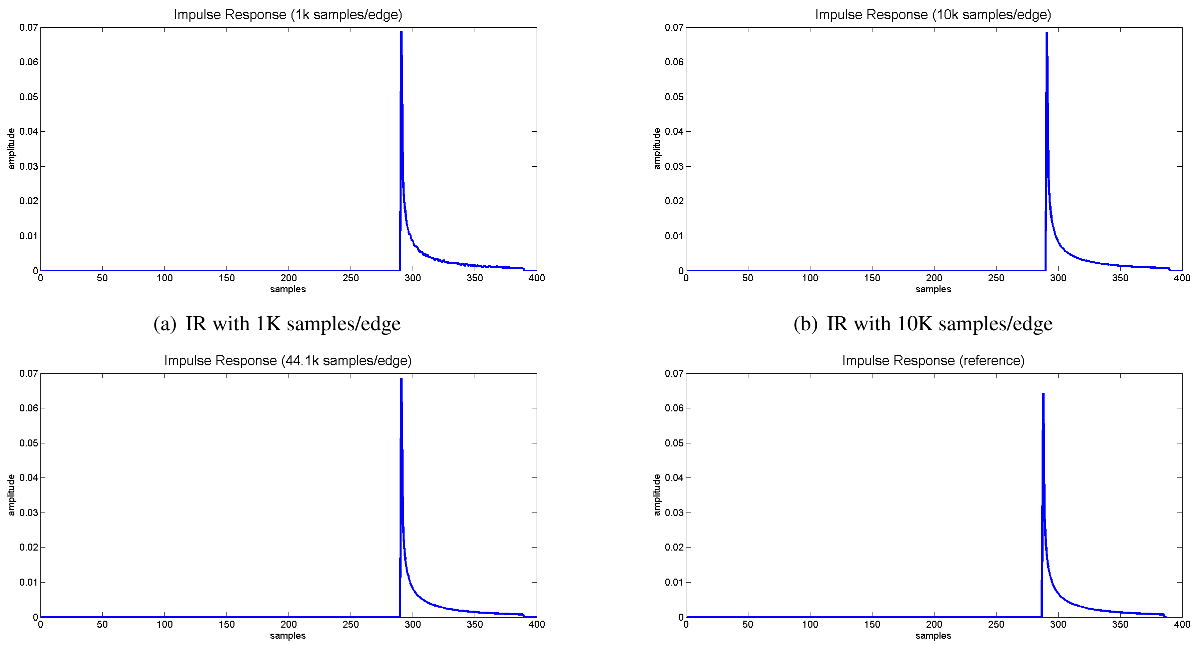
ANTONACCI, F., FOCO, M., SARTI, A., AND TUBARO, S. 2004. Fast modeling of acoustic reflections and diffraction in complex environments using visibility diagrams. In *Proceedings of 12th European Signal Processing Conference (EUSIPCO '04)*, 1773–1776.

BIOT, M. A., AND TOLSTOY, I. 1957. Formulation of wave propagation in infinite media by normal coordinates with an application to diffraction. In *Ocean Acoustics: Theory and Experiment in Underwater Sound*. Acoustical Society of America, 305–330.

BITTNER, J., MATTAUSCH, O., WONKA, P., HAVRAN, V., AND WIMMER, M. 2009. Adaptive global visibility sampling. In *SIGGRAPH '09: ACM SIGGRAPH 2009 papers*, ACM, New York, NY, USA, 1–10.

CALAMIA, P. T., AND SVENSSON, U. P. 2007. Fast time-domain edge-diffraction calculations for interactive acoustic simulations. *EURASIP J. Appl. Signal Process. 2007*, 1, 186–186.

CALAMIA, P. T., SVENSSON, U. P., AND FUNKHOUSER, T. A. 2005. Integration of edge diffraction calculations and geometrical-acoustics modeling. In *Proceedings of Forum Acusticum 2005*.



(a) IR with 1K samples/edge

(b) IR with 10K samples/edge

(c) IR with 44K samples/edge

(d) Reference IR computed by MATLAB toolbox

Figure 15: *Accuracy of impulse responses computed by our system for first order diffraction about a single finite wedge. Parts (a)–(c) show the variation in the IR with increasing number of samples per edge. As the sampling increases, the IR approaches the reference IR computed by the MATLAB toolbox, shown in part (d).*

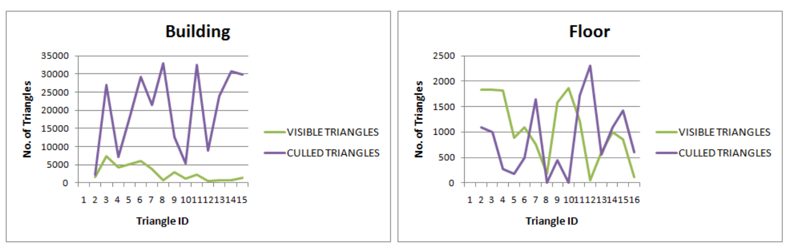


Figure 16: *Number of triangles culled by our occluder selection and occlusion culling steps for various benchmarks, as compared to the visible set size. Note that we use a FastV frustum subdivision of 4x4 and a quad subdivision of 4x4.*

CHANDAK, A., LAUTERBACH, C., TAYLOR, M., REN, Z., AND MANOCHA, D. 2008. Ad-frustum: Adaptive frustum tracing for interactive sound propagation. *IEEE Transactions on Visualization and Computer Graphics 14*, 6, 1707–1722.

CHANDAK, A., ANTANI, L., TAYLOR, M., AND MANOCHA, D. 2009. Fastv: From-point visibility culling on complex models. *Computer Graphics Forum 28*, 1237–1246(10).

CHHUGANI, J., PURNOMO, B., KRISHNAN, S., COHEN, J., VENKATASUBRAMANIAN, S., JOHNSON, D. S., AND KUMAR, S. 2005. vLOD: High-fidelity walkthrough of large virtual environments. *IEEE Transactions on Visualization and Computer Graphics 11*, 1, 35–47.

CISKOWSKI, R., AND BREBBIA, C. 1991. *Boundary Element methods in acoustics*. Computational Mechanics Publications and Elsevier Applied Science.

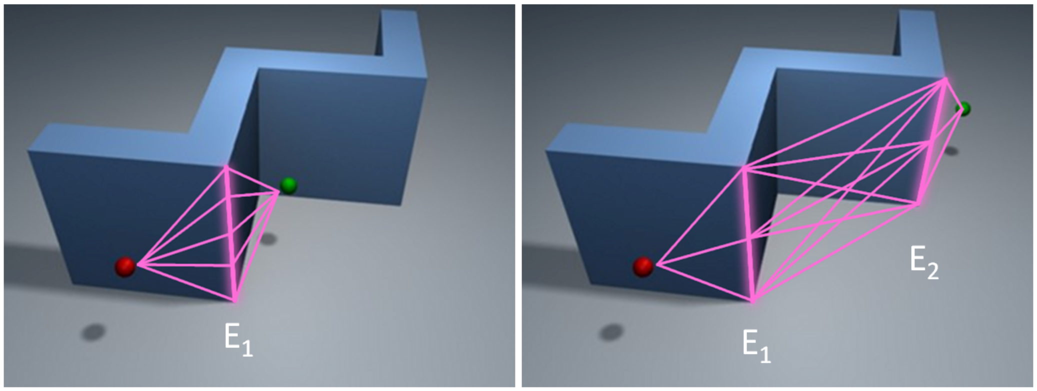
COORG, S., AND TELLER, S. 1997. Real-time occlusion culling for models with large occluders. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, ACM, New York, NY, USA, 83–ff.

DALENBÄCK, B. 1996. Room acoustic prediction based on a unified treatment of diffuse and specular reflection. *The Journal of the Acoustical Society of America 100*, 2, 899–909.

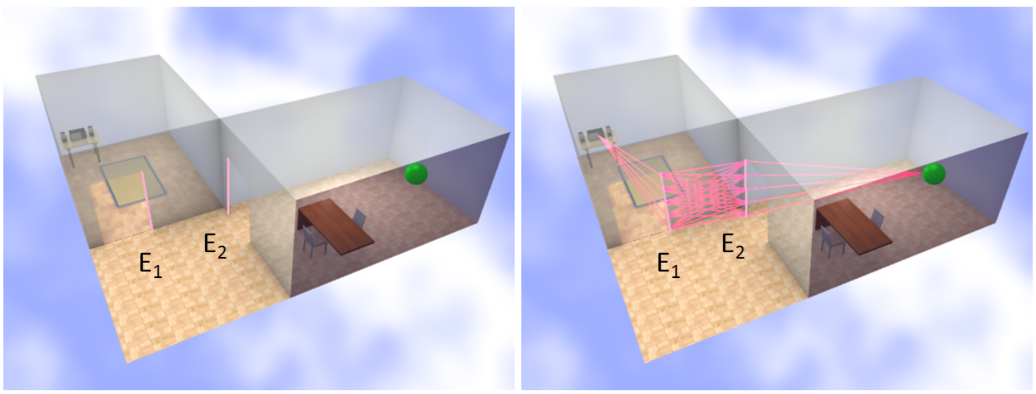
DURAND, F., DRETTAKIS, G., AND PUECH, C. 1996. The 3d visibility complex: a new approach to the problems of accurate visibility. In *Proceedings of the eurographics workshop on Rendering techniques '96*, Springer-Verlag, London, UK, 245–256.

DURAND, F., DRETTAKIS, G., AND PUECH, C. 1997. The visibility skeleton: a powerful and efficient multi-purpose global visibility tool. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 89–100.

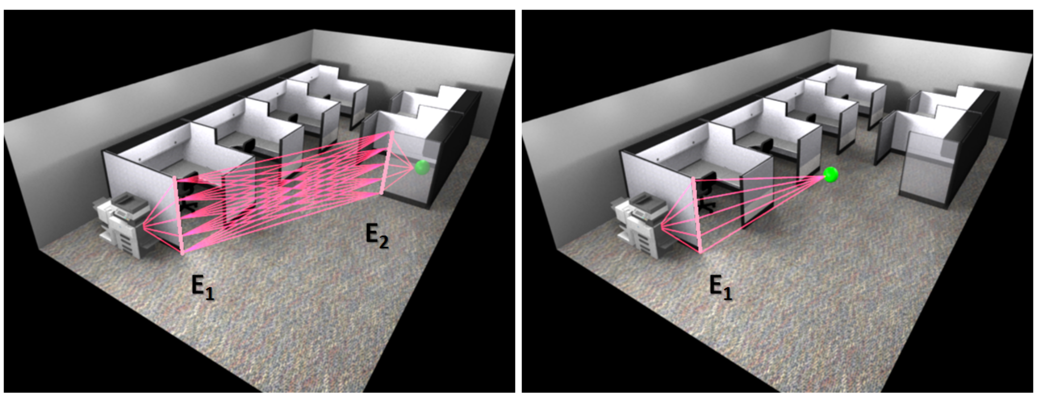
DURAND, F., DRETTAKIS, G., THOLLOT, J., AND PUECH, C. 2000. Conservative visibility preprocessing using extended projections. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 239–248.



(a) double wedge, first order (b) double wedge, second order



(c) house (d) house, second order



(e) office, second order (f) office, first order

Figure 17: *Some examples of diffraction paths computed by our algorithm. Parts (a) and (b) show first and second order diffraction paths, respectively, around a wall shaped like a double wedge. Parts (c) and (d) show the House scene and a second order diffraction path in it, respectively. Parts (e) and (f) show second and first order diffraction paths, respectively, in an Office scene. In each case, diffracting edges are highlighted and labeled; for second order paths E_1 is the first edge encountered along the path from source to listener, and E_2 is the second edge encountered. In each case, the listener is indicated by a green sphere, and the source is indicated by a red sphere (Parts (a) and (b)), the speakers (Parts (c) and (d)) or the printer (Parts (e) and (f)).*

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.

GIGUS, Z., CANNY, J., AND SEIDEL, R. 1991. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Trans. Pattern Anal. Mach. Intell. 13*, 6, 542–551.

HUDSON, T., MANOCHA, D., COHEN, J., LIN, M., HOFF, K., AND ZHANG, H. 1997. Accelerated occlusion culling using shadow frusta. In *SCG '97: Proceedings of the thirteenth annual symposium on Computational geometry*, ACM, New York, NY, USA, 1–10.

KAPRALOS, B., JENKIN, M., AND MILIOS, E. 2004. Sonel mapping: acoustic modeling utilizing an acoustic version of photon mapping. In *In IEEE International Workshop on Haptics Audio Visual Environments and their Applications (HAVE 2004)*, 2–3.

KOLTUN, V., AND COHEN-OR, D. 2000. Selecting effective occluders for visibility culling. In *Eurographics Short Presentations*.

KOUYOUMJIAN, R. G., AND PATHAK, P. H. 1974. A uniform geometrical theory of diffraction for an edge in a perfectly conducting surface. *Proc. of IEEE 62* (Nov.), 1448–1461.

KROKSTAD, A., STROM, S., AND SORSDAL, S. 1968. Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration 8*, 1 (July), 118–125.

KUNZ, K., AND LUEBBERS, R. 1993. *The Finite Difference Time Domain for Electromagnetics*. CRC Press.

LAINE, S., SILTANEN, S., LOKKI, T., AND SAVIOJA, L. 2009. Accelerated beam tracing algorithm. *Applied Acoustic 70*, 1, 172–181.

LAUTERBACH, C., CHANDAK, A., AND MANOCHA, D. 2007. Adaptive sampling for frustum-based sound propagation in complex and dynamic environments. In *Proceedings of the 19th International Congress on Acoustics*.

LEHTINEN, J., 2003. Time-domain numerical solution of the wave equation.

MEDWIN, H., CHILDS, E., AND JEBSEN, G. M. 1982. Impulse studies of double diffraction: A discrete huygens interpretation. *The Journal of the Acoustical Society of America 72*, 3, 1005–1013.

MONK, P. 2003. *Finite Element Methods for Maxwell's Equations*. Oxford University Press.

NIRENSTEIN, S., BLAKE, E., AND GAIN, J. 2002. Exact from-region visibility culling. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 191–202.

PULKKI, V., LOKKI, T., AND SAVIOJA, L. 2002. Implementation and visualization of edge diffraction with image source method. In *Proceedings of the 112th AES Convention*.

RAGHUVANSHI, N., GALOPPO, N., AND LIN, M. C. 2008. Accelerated wave-based acoustics simulation. In *SPM '08: Proceedings of the 2008 ACM symposium on Solid and physical modeling*, ACM, New York, NY, USA, 91–102.

SCHRÖDER, D., AND LENTZ, T. 2006. Real-Time Processing of Image Sources Using Binary Space Partitioning. *Journal of the Audio Engineering Society (JAES) 54*, 7/8 (July), 604–619.

SCHRÖDER, D., AND POHL, A. 2009. Real-time hybrid simulation method including edge diffraction. In *EAA Auralization Symposium*.

SILTANEN, S., LOKKI, T., KIMINKI, S., AND SAVIOJA, L. 2007. The room acoustic rendering equation. *The Journal of the Acoustical Society of America 122*, 3 (September), 1624–1635.

SILTANEN, S., LOKKI, T., AND SAVIOJA, L. 2009. Frequency domain acoustic radiance transfer for real-time auralization. *Acta Acustica united with Acustica 95*, 106–117(12).

SVENSSON, U., AND CALAMIA, P. 2006. Edge-diffraction impulse responses near specular-zone and shadow-zone boundaries. In *Acta Acustica united with Acustica*, vol. 92, 501–512.

SVENSSON, P., AND KRISTIANSEN, R. 2002. Computational modelling and simulation of acoustic spaces. In *22nd International Conference: Virtual, Synthetic, and Entertainment Audio*.

SVENSSON, U. P., FRED, R. I., AND VANDERKOOY, J. 1999. An analytic secondary source model of edge diffraction impulse responses . *Acoustical Society of America Journal 106* (Nov.), 2331–2344.

SVENSSON, P., 1999. Edge diffraction toolbox for matlab. http://www.iet.ntnu.no/ svensson/Matlab.html.

TAYLOR, M., CHANDAK, A., REN, Z., LAUTERBACH, C., AND MANOCHA, D. 2009. Fast edge-diffraction for sound propagation in complex virtual environments. In *EAA Auralization Symposium*.

TELLER, S. J., AND SÉQUIN, C. H. 1991. Visibility preprocessing for interactive walkthroughs. *SIGGRAPH Comput. Graph. 25*, 4, 61–70.

TELLER, S. J. 1992. Computing the antipenumbra of an area light source. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, ACM, New York, NY, USA, 139–148.

TSINGOS, N., FUNKHOUSER, T., NGAN, A., AND CARLBOM, I. 2001. Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proc. of ACM SIGGRAPH*, 545–552.

VORLANDER, 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.

WONKA, P., WIMMER, M., ZHOU, K., MAIERHOFER, S., HESINA, G., AND RESHETOV, A. 2006. Guided visibility sampling. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, ACM, New York, NY, USA, 494–502.