Proceedings of the International Symposium on Room Acoustics, ISRA 2010

29–31 August 2010, Melbourne, Australia

Fast and Accurate Geometric Sound Propagation using Visibility Computations

Anish Chandak (1), Lakulish Antani (1), Micah Taylor (1), and Dinesh Manocha (1)

(1) University of North Carolina at Chapel Hill, Chapel Hill, U.S.A
http://gamma.cs.unc.edu/research/sound/

PACS: 43.55.Ka – Computer simulation of acoustics in enclosures, modeling; 43.58.Ta – Computers and computer programs in acoustics

ABSTRACT

Geometric Acoustics (GA) techniques based on the image-source method, ray tracing, beam tracing, and ray-frustum tracing, are widely used to compute sound propagation paths. In this paper, we highlight the connection between these propagation techniques with the research on visibility computation in computer graphics and computational geometry. We give a brief overview of visibility algorithms and apply some of these methods to accelerate GA, specifically early specular reflections and finite-edge diffraction. Moreover, we survey our recent work on fast and accurate GA methods that use accurate and conservative visibility techniques. This includes: a) an algorithm for fast computation of early specular reflections using conservative from-point visibility computation; and b) a fast method for finite-edge diffraction using conservative from-region visibility computation. Our approach for computing specular reflections is based on the image-source method and we reduce the number of image sources by using conservative visibility computations. The edge diffraction computation is based on the well known Biot-Tolstoy-Medwin (BTM) diffraction model and we combine it with efficient algorithms for region-based visibility to significantly reduce the number of edge pairs that need to be processed for higher-order diffraction computation. We highlight the performance of these methods on many complex models. Our initial results indicate that we obtain considerable speedups over prior methods for accurate geometric sound propagation.

INTRODUCTION

Geometric acoustic (GA) algorithms are commonly used to compute sound propagation paths for room acoustics, virtual reality (VR), games, and visualization. The earliest work in this area is that of Krokstad et al. [1] based on ray tracing, as well as the image-source method [2]. During the last decade, the major trend in GA has been on developing faster algorithms that can handle complex 3D models represented using tens of thousands of *geometric primitives* (i.e. triangles, planes, edges, etc.) and compute the propagation paths from a source to a listener based on reflections and edge diffraction. Furthermore, interactive applications like games and VR applications need the capability to perform these computations at 20 frames per second (or higher) on commodity hardware.

In this paper, we give an overview of recent research on visibility computation in computer graphics and related areas. Given a 3D model, which is represented using geometric primitives, the goal of visibility algorithms is to compute the visible primitives from a given view-point or view-region in 3D. We show that many GA methods used for computing specular reflections and edge diffraction can be accelerated by using visibility algorithms. Most of the work in computer graphics has focused on computing sample-based visibility at the resolution of the final image and current GPUs (graphics processors) can perform these computations very fast. In contrast, accurate computations of propagation paths needs the capability to perform visibility computations at the precision of the original 3D model or *object-space visibility*. In this regard, we give a brief overview of two recent object-space visibility algorithms which have been used for fast GA computations. This includes a point-based conservative visibility algorithm [3] that can improve the computation of specular reflections using image-source method. The second algorithm [4] computes visibility from a given edge in the model and is used to accelerate the performance of higher order finite-edge diffraction based on the Biot-Tolstoy-Medwin (BTM) diffraction model [5, 6]. We also highlight the speedups obtained by these visibility algorithms on complex models.

VISIBILITY TECHNIQUES

Visibility techniques have been extensively studied in computer graphics, computational geometry, robotics and related areas for more than four decades. We refer the readers to excellent recent surveys [7, 8] for a comprehensive overview of visibility techniques. In this section, we give a brief overview.

The basic goal of visibility algorithms is to compute a set of primitives that are visible from a given view-point or view-region. Visibility algorithms can be classified in different ways. One way is to classify them into from-point and from-region visibility. Figure 1a shows an example of from-point visibility where the circle represents the view-point. From-point visibility is extensively used in computer graphics for generating the final image from the eye-point based on rasterization [9] or ray tracing [10]. Other examples of applications of from-point visibility include hard shadow computation for point light sources. Figure 1d shows an example of from-region visibility where the rectangular region with the arrows denotes a view-region. From-region visibility has been used extensively in computer graphics for global illumination (i.e., computing the multiple

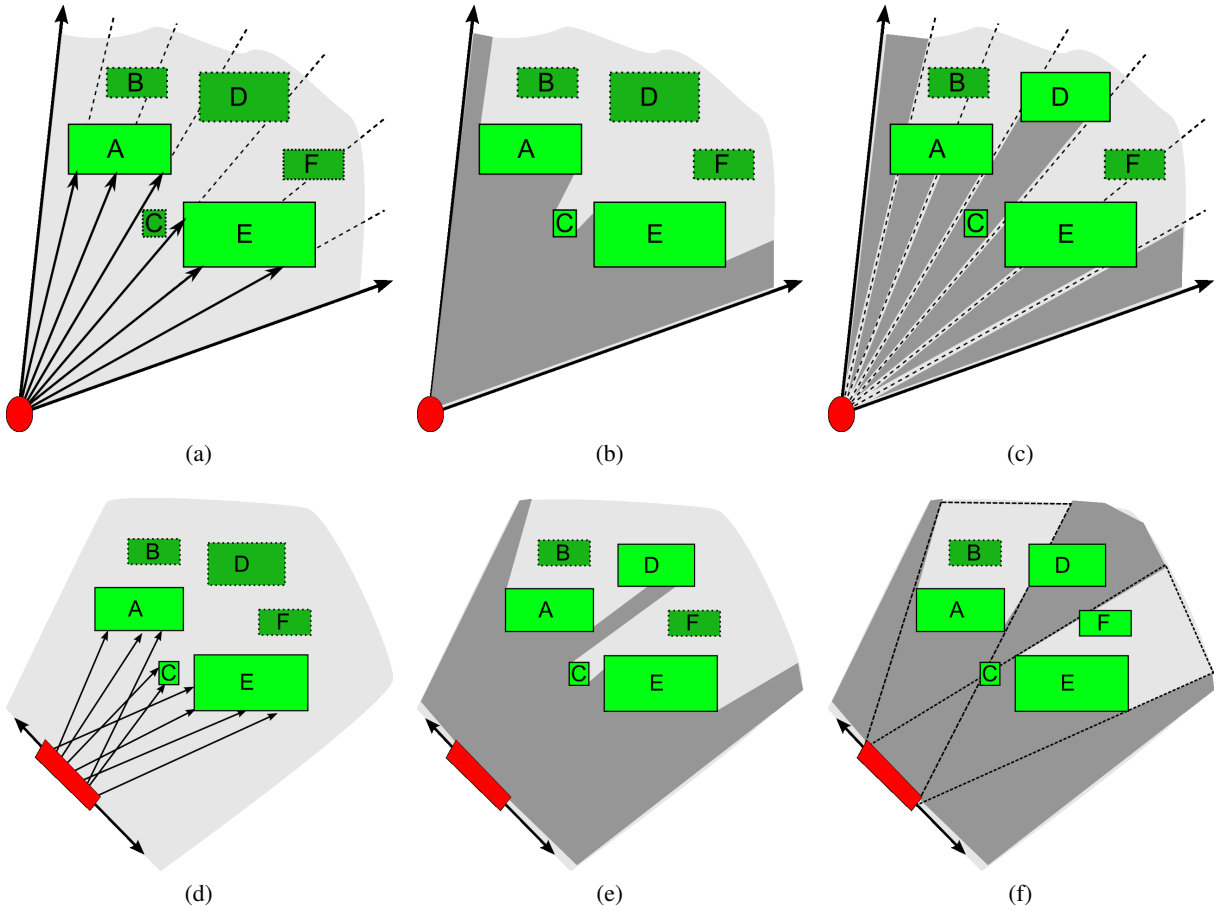


Figure 1: Classification of visibility algorithms: (a) Sample-based from-point visibility. The visibility computation is accurate up to the resolution of the rays used. (b) Exact from-point visibility, based on object-space computations. (c) Conservative from-point visibility, which tends to compute a superset of primitives that are visible from a given view-point. (d) Sample-based from-region visibility. (e) Exact from-region visibility. (f) Conservative from-region visibility. The red circle and red rectangle denote a view-point and a view-region respectively. The light gray region bounded by the two arrows is the viewing frustum that is used to compute the visible primitives. The geometric primitives are labeled A, B, C, D, E, and F. The visible primitives are marked as solid bright green boxes. The dark gray region is the region consisting of visible primitives as determined by the visibility algorithm. In (c) the visible region is determined by shooting frusta [3] and in (e) the visible region is determined by constructing shadow frusta for primitives A and E [4].

bounce response of light from light sources to the camera via reflections from primitives in the 3D model), interactive walk-throughs of complex 3D models (by prefetching a smaller set of potentially visible primitives from a region around the active camera position), soft shadow computation from area light sources, etc.

We now formally define from-point and from-region visibility, and will them in later sections to describe an efficient image-source method for specular reflection and finite-edge diffraction computation. Given a view-point ($\mathbf{v} \in \Re^3$, from-point) or a view-region ($\mathbf{v} \subset \Re^3$, from-region), a set of geometry primitives (Π), and a viewing frustum (Φ), which is a set of infinitely many rays originating in $\mathbf{v}$, the goal of visibility techniques is to compute a set of primitives $\pi \subseteq \Pi$ hit by rays in Φ. For example, in Figure 1a the red circle corresponds to the view-point and in Figure 1d the red rectangle corresponds to the view-region. The set of primitives is $\Pi = \{A,B,C,D,E,F\}$ and the region shaded in light gray bounded by two arrows is spanned by rays in Φ, the viewing frustum. In Figure 1a the visible set of primitives $\pi = \{A,E\}$ and in Figure 1d the visible set of primitives $\pi = \{A,C,E\}$. Note that the set π is called the potentially visible set (PVS). Depending on the properties of the computed PVS, visibility techniques can be further classified.

Object-Space Exact Visibility

This class of visibility techniques computes a PVS, π_{exact}, hit by every ray in Φ and every primitive in π_{exact} is hit by some ray in Φ. Since every ray in Φ is considered to compute visibility, these techniques are called object-space techniques. Moreover, these intersection computations are performed at the accuracy of the original model, e.g. IEEE 64-bit double precision arithmetic. The PVS computed by an object-space exact visibility algorithm is the smallest PVS which contains all the primitives visible from $\mathbf{v}$. Many applications require exact visibility with object-space precision. For example, accurate computation of soft shadows due to area light source in computer graphics [11] requires the computation of exact visible area from all the points of the area light source to compute the contribution of the area light source at the point. Similarly, computing hard shadows due to a point light source requires accurate computation of visible portions of primitives from the point light source [12] or aliasing artifacts may appear.

From-Point Visibility: Figure 1b shows an example of exact from-point visibility. Primitives A, C, and E block all the rays in the viewing frustum starting at the view-point from reaching the primitives B, D, and F. Thus, the primitives B, D, and F are marked as hidden. The two main approaches for computing exact from-point visibility are based on beam tracing [13, 14]

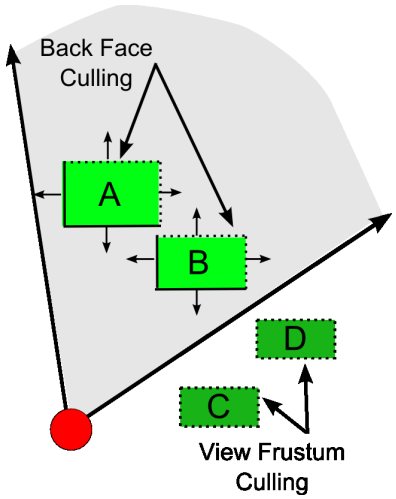


Figure 2: View-frustum culling and back-face culling to trivially compute hidden primitives. In practice, these algorithms are easier to implement as compared to advanced culling methods but are highly conservative and do not find many hidden primitives.

and Plücker coordinates [15].

Beam tracing approaches shoot a beam from the view point and perform exact intersections of the beam with the primitives in the scene. As the beam hits the primitives, exact intersection and clipping computations are performed between the beam and the primitive. The portion of the beam which is not hit by any primitive so far is checked for intersections with the remaining primitives. Thus, the complexity of the shape of the beam may increase as more intersection computations are performed. In general, performing exact and robust intersection computations with the beam on complex 3D models is considered a hard problem.

Approaches based on Plücker coordinates perform constructive solid geometry (CSG) operations in Plücker space to compute exact visibility. Plücker space is a six-dimensional space with certain special properties [16]. In this approach, the view-frustum and the primitives are represented in Plücker space as CSG primitives and intersection computations are performed between the view-frustum and the primitives such that when the CSG intersection is transformed back into Euclidean space, it corresponds exactly to the visible primitives. The intersection between the view-frustum and primitives in Plücker space requires complex operations. Thus, these techniques can be used to perform exact from-point visibility computations, but can be expensive and susceptible to robustness problems.

From-Region Visibility: Figure 1e shows an example of from-region exact visibility. Primitives A, C, D, and E are visible from the view-region. Note that no ray starting in the view region reaches B and F, and therefore they are marked as hidden from the view-region. Many complex data structures and algorithms have been proposed to compute exact from-region visibility, including aspect graphs [17], visibility complex [18, 19], and performing CSG operations in Plücker space [16]. 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 geometry primitives – and are too slow to be of practical use on complex models.

Object-Space Conservative Visibility

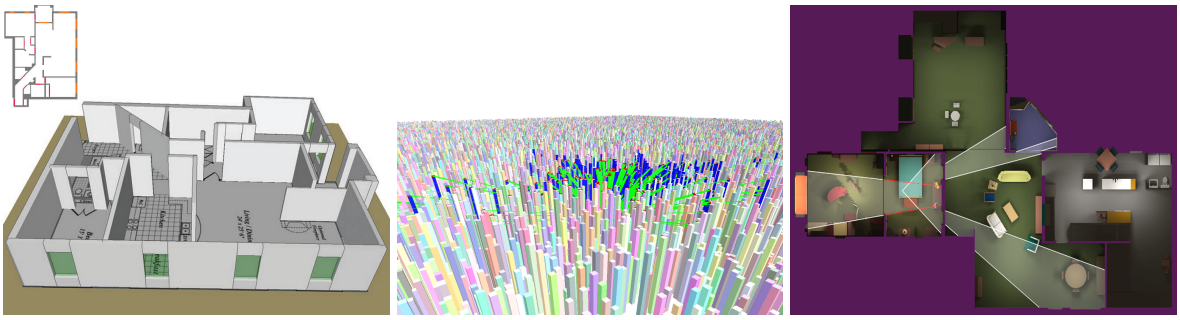
These visibility techniques compute a PVS, $\pi_{conservative}$, hit by at least every ray in Φ. The PVS, $\pi_{conservative}$, may contain primitives which are not hit by any ray in Φ. Thus, $\pi_{conservative}$ is conservative, i.e., $\pi_{conservative} \supseteq \pi_{exact}$. Conservative from-point visibility algorithms are preferred for their computational efficiency and simplicity over exact algorithms. The two simple and widely used but highly conservative visibility techniques are view-frustum culling and back-face culling. They are used to trivially compute some of the hidden primitives. Figure 2 illustrates these methods. In view-frustum culling, the primitives completely outside the view-frustum are marked hidden; and in back-face culling, the primitives which are facing away from the view-point or view-region are marked as hidden. Conservative visibility is preferred in many applications mainly due to its ease of implementation and good performance improvement. The choice between a conservative or an exact object-space algorithm is decided by the application on the basis of the trade-off between the overhead of extra visible primitives due to the conservative algorithm vs. the time overhead of the exact algorithm.

From-Point Visibility: In Figure 1c we show an example of the PVS computed by our conservative from-point visibility algorithm (FastV) [3]. Many small frusta are shot from the view-point and a frustum stops when it is entirely blocked by primitives. Note that primitive D, which is not visible from the view-point, is still reported as potentially visible by our conservative approach. Primitives B and F remain hidden from the view-point. Many other techniques have been developed for conservative from-point visibility computations [22, 23, 24, 25, 26]. Many of these algorithms have been designed for special types of models, e.g. architectural models represented as cells and portals, 2.5D urban models, or scenes with large convex primitives. These methods are well suited when the target application of the visibility algorithms is limited to urban scenes or architectural models corresponding to buildings or indoor structures with no interior primitives or furniture. Figure 3 gives examples of these special kinds of models that can be handled by prior methods. In contrast, our FastV algorithm [3] is general and can handle all kinds of scenes, as shown in Figure 8.

From-Region Visibility: Figure 1f demonstrates our conservative from-region visibility algorithm [4]. The basic idea is to construct *shadow frusta* (polyhedral beams contained within the *umbrae* between the view-region and primitives) for selected primitives. Typically, these primitives are selected by an *occluder selection* algorithm based on their effectiveness in removing hidden primitives. Primitives which are completely inside the shadow frusta are marked as hidden. In Figure 1f, only the primitive B is completely inside the shadow frusta of primitives A and E. Also, note that primitive F is marked as potentially visible by our approach even though there is no ray originating in the view-region which reaches F. Many other algorithms have also been proposed for conservative from-region visibility. Several algorithms exist for performing occlusion culling with respect to shadow frusta [27, 28], with different trade-offs and limitations. Some conservative algorithms operate in the dual space of rays, by dividing the scene into *cells* separated by *portals* [29] and computing *stabbing lines* [30] through portals.

Image-space or Sample-based Visibility

These approaches sample the set of rays in Φ and compute a PVS, $\pi_{sampling}$, that is hit by only the finite set of sampled rays. Note that since $\pi_{sampling}$ is computed for only a finite subset of rays in Φ, $\pi_{sampling} \subseteq \pi_{exact}$. The choice of sampled rays is governed by the application. Sampling-based methods are widely used in graphics applications due to their computational efficiency and are well supported by current GPUs. Typically, during image generation, an image of a given resolution, say 1K $\times$ 1K pixels and only a constant number of rays per pixel are sampled to generate an image. Sampling based methods are extensively used in computer graphics for image generation. However, these methods can suffer from spatial and temporal



(a) (b) (c)

Figure 3: Special scenes used for visibility computation: (a) Buildings with clearly marked cells and portals and no geometry or furniture inside the cells (source [20]). (b) Urban scenes, which can be represented using 2.5D objects or height fields (source [21]). (c) Simple scene with large occluders (source [22]). The walls of the rooms are used as occluders.

aliasing issues and may require supersampling or other techniques (e.g. filters) to reduce aliasing artifacts.

From-Point Visibility: We show an example of from-point sample-based visibility in Figure 1a. Only a few rays are sampled and intersected with the geometric primitives to find the visible primitives. This could lead to spatial aliasing, as shown in Figure 1a. The primitive C is marked as hidden because it lies between two sampled rays even though it is visible from the view-point. Despite their short comings sample-based methods are widely used in computer graphics [31]. Efficient implementation of sample-based visibility algorithms can be achieved on current graphics processing units (GPUs) [32]. The z-buffer algorithm [9] is a standard sample-based visibility algorithm that is supported by the rasterization hardware in GPUs. Moreover, advanced support for sample-based visibility, such as from-point occlusion queries are also supported in GPUs [33]. Also, sample-based ray shooting techniques have been used in computer graphics [10].

From-Region Visibility: We show an example of from-region sample-based visibility in Figure 1d. Similar to from-point visibility, the sampling in from-region algorithms introduces spatial aliasing. In this case, the primitive D is marked as hidden even though there exists at least one ray from the view-region that reaches the primitive D. However, these methods are fast compared to exact and conservative from-region visibility algorithms and can easily be applied to complex models [34, 35]. However. they have one important limitation: they sample a finite set of rays originating inside the view-region and thus compute only a *subset* of the exact solution (i.e., approximate visibility). Therefore, these methods are limited to sampling-based applications such as interactive graphical rendering, and may not provide sufficient accuracy for applications where an accurate from-region solution is needed.

GEOMETRIC ACOUSTICS

In this section, we describe new geometric sound propagation algorithms based on recently developed object-space conservative from-point [3] and from-region [4] visibility techniques. Our geometric sound propagation algorithms are based on the image-source method [36, 37]. As originally formulated, the image-source method can mainly simulate specular reflections. However, it is possible to extend this method to handle edge diffraction by introducing line or edge image sources [38, 39, 40, 41]. First, we present an overview of the image-source method based on the *visibility tree* and use our object-space visibility algorithms to accelerate the construction of the *visibility tree*. Next, we provide a brief description of the computation of the visibility tree for specular reflection and finite-edge diffraction. For a detailed description of our technique we refer the readers to other papers [3, 4].

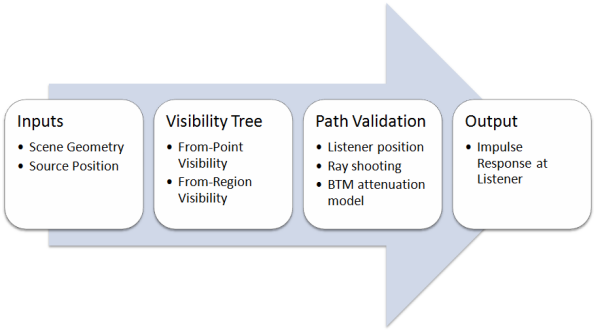
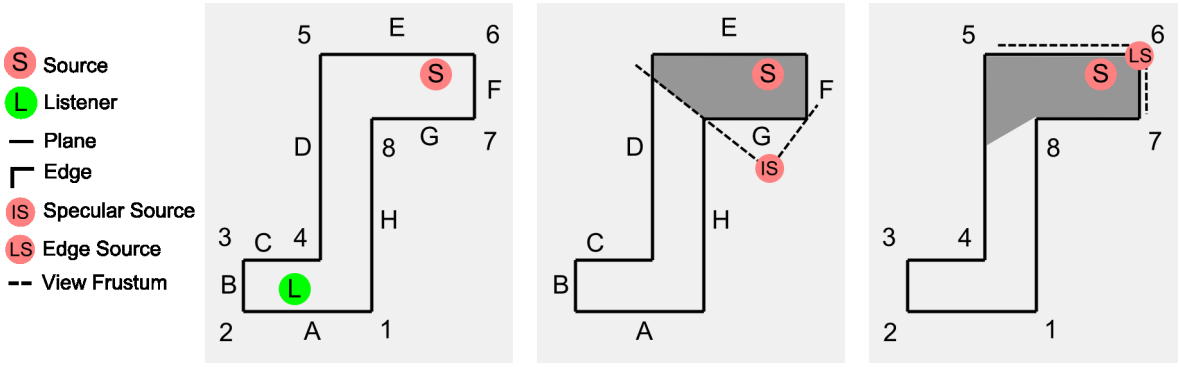


Figure 4: Overview of our modified image-source sound propagation algorithm based on the visibility tree. Using the scene geometry and source position as the input, we first construct a visibility tree that represents the potential propagation paths. Next, we use the listener position to find valid propagation paths using the visibility tree. Finally, we use the valid paths to compute the impulse response at the listener.

Overview

Given a point sound source, the CAD model with acoustic material properties, diffracting edges, and the listener position, our goal is to compute an impulse response (IR) of the acoustic space for the source and listener position. The IRs can be used to derive various acoustic parameters of a room. In Figure 5a, a CAD model (shown in top down view) consists of specular planes A to H and diffracting edges 1 to 8. The source and listener positions are also shown.

To compute the IR, we need to compute all the specular and diffraction propagation paths that reach the listener from the source. We use a two-step approach based on the image-source method [42] (see Figure 4). In the first step we construct a *visibility tree* $VT(S,k)$ from a source S up to a user-specified k orders of reflection. Note that we only need to compute image sources for a source (or image source) S with respect to the triangles and/or edges that are visible to S. If S is a point source, this involves from-point visibility computation. For example, consider the image source, IS of the source S about plane G in Figure 5b; we need to compute only the image sources of IS about planes D, E, and F for second order specular reflection from IS. If S is a line or an edge source, however, we need to perform from-region visibility computation, specifically, from-edge visibility computation, which computes a superset of all the primitives that can be visible from any point on the edge.



(a) (b) (c)

Figure 5: Accelerating image-source methods using visibility computations: (a) A top down view of a simple configuration with plane primitives A-H and edge primitives 1-8. The source and the listener are also shown. (b) Consider the image source corresponding to the source due to specular reflection from Plane G. Only the planes visible from the image source IS shown needs to be considered to compute second order reflection through plane G. (c) The line image source due to diffraction from edge 6 is shown. After computing from-region visibility from line source 6 only a subset of other edges needs to be considered for second-order edge diffraction.

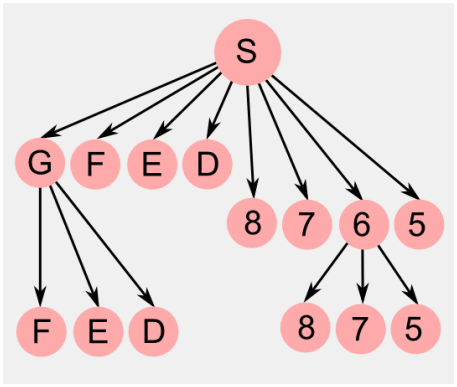


Figure 6: Visibility tree for the configuration shown in Figure 5 up to two orders of specular reflection and edge diffraction. Note that second order image sources have been constructed for the image source from plane G for specular reflection and the line image source from edge 6 for edge diffraction.

For example, second order diffraction about the line source LS in Figure 5c can only occur with edges 5, 7, and 8.

The visibility algorithms are applied recursively for point and line image sources to construct the visibility tree. An example visibility tree for the configuration in Figure 5 is shown in Figure 6. Each path in $VT(S,k)$ represents a potential path contributing to the IR. Each path consists of a sequence of (up to k) triangles and/or edges that a ray starting from S reflects and/or diffracts about as it reaches the listener at the position L. For example, $S \rightarrow G \rightarrow E$ denote all specular paths from the source that bounces off planes G and then E. Similarly, $S \rightarrow 6 \rightarrow 7$ denote all diffraction paths from the source that hit edge 6 and then edge 7. However, to compute the final path from visibility tree, the listener position is required. Thus, in the second step, given a listener position L, we attach a listener node to every node in the tree and for each potential path in $VT(S,k)$, we determine which of the propagation paths are valid. Thus, validating $S \rightarrow G \rightarrow E \rightarrow L$ means finding a specular path from source that bounces off plane G and then plane E and then reaches the listener (Figure 7a). Similarly, validating $S \rightarrow 6 \rightarrow 7 \rightarrow L$ means finding multiple paths from the source that hit edge 6 followed by edge 7 and then reach the listener (Figure 7b). It is possible that some of the paths are blocked by other primitives in the scene and may not contribute to the IR. We refer to the second step as *path validation*.

Image Source Method

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 basically 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, and this direct path also passes through T. 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 by applying from-point visibility techniques [3]. 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.

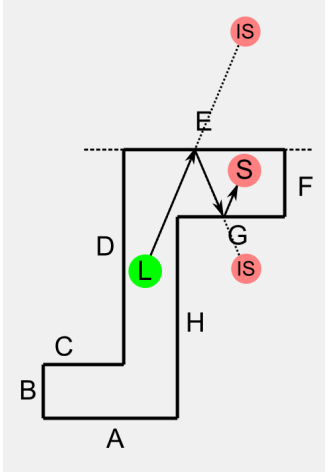
BTM based Finite-Edge Diffraction

We now briefly outline a method of integrating edge diffraction modeling into the image source method [40]. Analogous to how specular reflection about a triangle is modeled by computing the image of the source with respect to the triangle, diffraction about an edge is modeled by computing the image of the source *with respect to the edge*. The key idea is that the image source from a point source S with respect to diffracting edge E is that edge E itself. This means that image sources can now be points or line segments. 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.

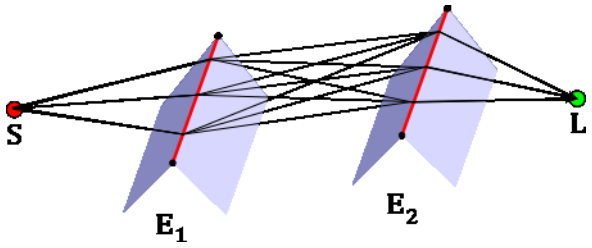
For a given edge source, the basic approach described above can be accelerated by applying from-region visibility techniques [4]. Note that second-order diffraction image sources for an edge source S_i need to be computed for edges that are visible from S_i. Also, specular reflections of S_i need to be computed from triangles that are visible from S_i.

RESULTS

We highlight some of the results of our novel sound propagation algorithm. Table 1 summarizes our results on early specular



(a)



(b)

Figure 7: (a) Path validation for specular reflection. Ray shooting tests are used to verify whether all segments of the path from source *S* to listener *L* are visible. (b) Path validation for second order finite-edge diffraction. Each edge is divided into small segments and ray shooting tests are performed from source to the edge segments, between edge segments, and edge segments to the listener to find visible paths from source *S* to listener *L*.

reflections. We also compare our results with Accelerated Beam Tracing (ABT) algorithm [42], which is another conservative from-point visibility algorithm. Our specular reflection results were obtained on models of complexity ranging from 438 triangles to 212K triangles. We also tested the performance on three benchmarks presented in [42] and compared the timings for constructing the *visibility tree* using our approach and ABT. We also used two additional complex benchmarks with 80K and 212K triangles. We are not aware of any implementation of the image source method that can handle models of such complexity in tractable time. We also compare the performance of our visibility algorithm (FastV, [3]) with the fastest beam tracing algorithm proposed by Overbeck et al. [14]. We chose a few camera frames for the Armadillo model (see Figure 8a) and compared the PVS computed by FastV with the PVS computed by the beam tracer. We observed that the size of the PVS computed by FastV converged to within $1-10\%$ of the exact from-point beam tracing PVS (see Figure 9). In terms of performance, FastV is about $5-8$ times faster on a single CPU core on the Armadillo model, as compared to [14].

Table 3 highlights the results on finite-edge diffraction. We compare the performance of our visibility tree construction step (using from-region visibility) against visibility tree construction using only the view-frustum culling in the MATLAB Edge Diffraction toolbox [43]. We compare the time required to build the visibility tree as well as the size of the tree constructed for each approach. Table 2 shows the breakdown of time spent in each step of our algorithm. It is evident from the table that the costliest step of our algorithm is the final IR computation as the path validation for edge diffraction requires shooting millions of rays. Constructing the visibility tree is much faster by comparison. Figure 10 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, visibility algorithms helps reduce the complexity of the visibility tree computed by our algorithm by a factor of 2 to 4.

Model	Tris	Time (sec)	Speed Up (ABT)
Room	438	0.16	10.1
Regular Room	1190	0.93	22.2
Complex Room	5635	6.50	11.8
Sibenik	78.2K	72.00	–
Trade Show	212K	217.60	–

Table 1: Early specular reflections: The performance of sound propagation algorithms for three orders of reflection on a single core. We observe $10-20$X speedup for the simple models over accelerated beam tracing (ABT) [42].

Scene	Visibility Tree (ms)	IR Computation (s)
Factory	141.0	23.9
Room	747.6	10.4
House	1045.6	24.3

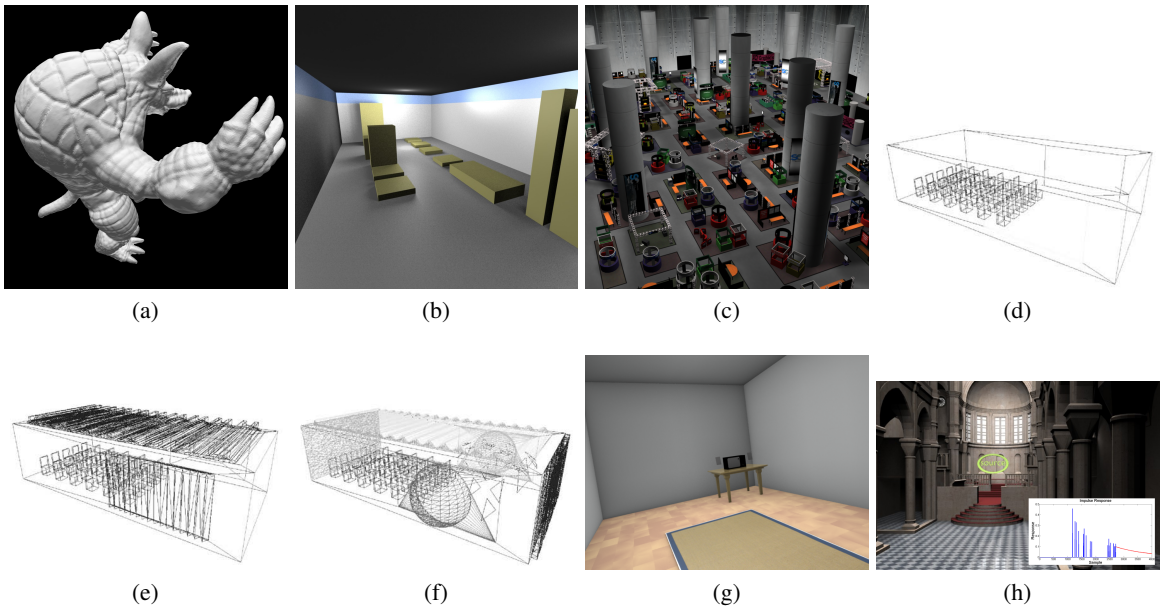
Table 2: Performance of individual steps of our algorithm. Column 2 shows the time spent in constructing the visibility tree, averaged over multiple source positions. Column 3 shows the time taken to compute the final IR, averaged over multiple source and listener positions.

GEOMETRIC ACOUSTICS AND VISIBILITY

In this section we present a discussion on different visibility algorithms for computing the *visibility tree* for early specular reflections and finite-edge diffraction. The choice of visibility algorithm depends on the target application. For instance, room acoustics software requires accurate modeling of early specular reflections and edge diffraction, therefore, exact or conservative object-space visibility algorithms are most suitable. Similarly, for entertainment applications like games it might be possible to use sample-based visibility algorithms as temporal and spatial aliasing issues can be hidden by applying heuristics which reduce the accuracy of the simulation.

Another example is that the cost of computing the diffraction paths and IRs for double or triple diffraction for finite-edge diffraction using the BTM model could be so high that it might be worth looking into exact visibility approaches to compute the smallest PVS from an edge and thus minimize the path validation steps. The exact visibility algorithms are relatively expensive and it is hard to implement them robustly in 3D. However, the savings in the size of the visible set may result in improved overall performance.

Sample-based Approaches: Due to their simplicity and efficiency, sampling based approaches are very popular in geometric acoustics [1]. But, the acoustic space has to be sampled densely to produce a robust solution. Since the sampling based approaches discretely sample the acoustic space, they introduce statistical errors [44] and may miss critical early reflection paths [45]. Many techniques like ray tracing [1, 46], ray-frustum tracing [47, 48], and other sample-based techniques [49, 50] have been applied to compute early specular reflection.



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

(e) (f) (g) (h)

Figure 8: Various benchmarks used in our work. (a) Armadillo (b) Factory (c) Trade Show (d) Room (source [42]) (e) Regular Room (source [42]) (f) Complex Room (source [42]) (g) House (h) Sibenik Cathedral.

Scene	Triangles	Edges	Second order diffraction paths in tree			Path Validation
			Our method	Toolbox	Size reduction	Speedup
Factory	170	146	4424	12570	2.84	1.93
Room	876	652	43488	181314	4.17	3.23
House	1105	751	133751	393907	2.95	13.74

Table 3: Columns 4–6 demonstrate the benefit of using from-region visibility to reduce second order diffraction paths between mutually invisible edges. The last column shows the speedup caused during path validation by this reduction in the size of the visibility tree.

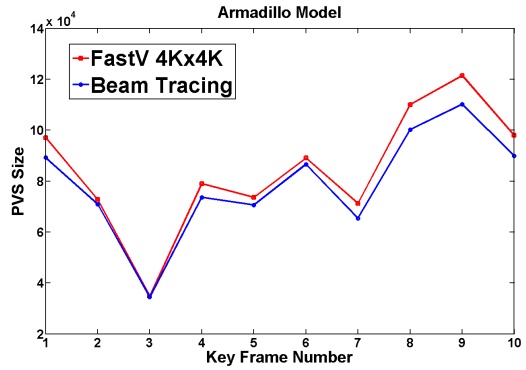
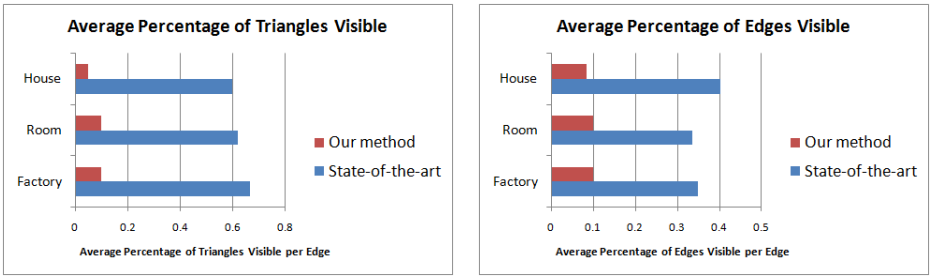


Figure 9: Comparison of the PVS computed by FastV and a beam tracer [14] for the Armadillo model.

We are not aware of any work on using sample-based from-region visibility algorithms to accelerate finite-edge diffraction. Some recent techniques for sample-based from-region algorithms [34, 35] can be applied on simple scenes but the impact of sampling needs to be carefully analyzed.

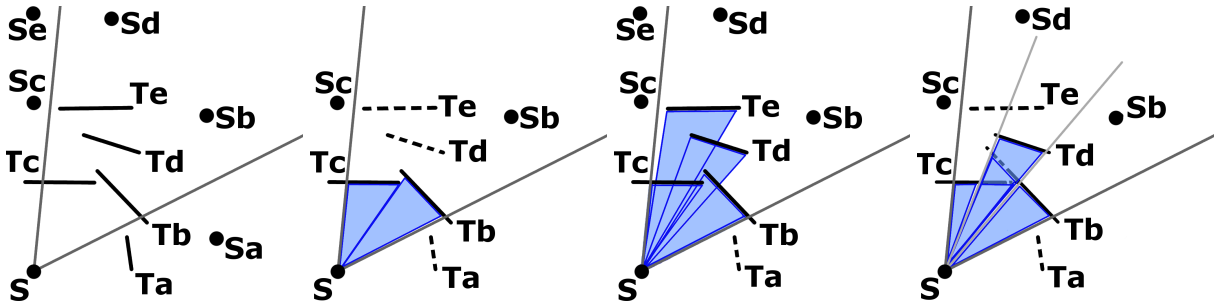
Object-Space Exact Approaches: The size of the visibility tree computed by exact object-space algorithms is guaranteed to be optimal. This improves the time taken by the *path validation* step since the number of potential paths to validate is the smallest. However, performing exact visibility to compute the visibility tree is compute intensive and may require a long time. Such methods have been applied for early specular reflection [51] for limited scenes with a cell-and-portal structure. Applying these algorithms for early specular reflections for general scenes is computationally expensive and requires a robust implementation. One possibility is to apply recently developed beam tracing algorithms [14] for early specular reflection. Like sample-based approaches, no known exact object-space from-region has been applied to improve the finite-edge diffraction computation. It is possible to apply aspect graphs [17], visibility complex [18, 19] to compute from-region visibility from a diffracting edge. However, the computational complexity of such methods – $O(n^9)$ for aspect graphs and $O(n^4)$ for the visibility complex, where n is the number of scene primitives – makes them impractical for even the simple scenes. Moreover, these are global visibility algorithms and compute visibility from all points in the scene; they cannot be used to compute visibility from a given list of diffracting edges.

Object-Space Conservative Approaches: Given the computational complexity of exact approaches and aliasing issues with sampling-based approaches, conservative approaches offer an interesting alternative. Conservative approaches have lower runtime complexity as compared to the exact approaches and do not suffer from the aliasing errors that are common in sample-based approaches. However, the PVS computed by conservative approaches is larger than that computed by exact or sample-based visibility approaches, therefore the size of visibility tree will be larger. Thus, the path validation step will take longer since there are more paths to validate. Figure 11 compares different image-source methods. The main difference between these methods is in terms of which image sources they choose to compute [42, 52]. A naïve image-source method computes image sources for all primitives in the scene [2]. Beam tracing methods compute the image sources for exactly visible primitives from



(a) (b)

Figure 10: Average amount of visible geometry returned by our from-region visibility approach as compared to the state-of-the-art [43] 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.



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

Figure 11: Geometric sound propagation approaches: Given a sound source, S, and primitives (a, b, c, d, and e) the image source method (see 11a) creates image-sources of S against all primitives in the scene. The beam tracing method (see 11b) computes image-sources for only exactly visible triangles, b and c in this case. The accelerated beam tracing approach computes image-sources for all triangles inside the beam volume (see 11c), i.e., b, c, d, and e in this case. Our approach (see 11d) computes image-sources for triangles b, c, and d.

a source (or image source). Methods based on beam tracing, like accelerated beam tracing [42], compute image sources for every primitive inside the beam volume. Our approach, shown in Figure 11d, finds a conservative PVS from a source and computes the image sources for the primitives in the conservative PVS. We have presented an approach based on a conservative from-point [3] and a conservative from-region [4] algorithm to compute early specular reflection and finite-edge diffraction. Accelerated Beam Tracing [42], a variant of beam tracing, has also been applied for early specular reflections. Regarding conservative visibility algorithms for finite-edge diffraction, only view-frustum culling has been applied [43] and our approach for reducing edge pairs for edge diffraction [4] is the only known implementation which uses visibility algorithms for finite edge diffraction.

SOUND SCATTERING

In the previous sections, we discussed accelerating early specular reflections and finite-edge diffraction by applying visibility techniques. However, only modeling specular reflections and finite-edge diffraction is insufficient to accurately predict the acoustics of an environment [53]. Sound scattering, i.e. interaction of sound waves with objects of size comparable to its wavelength, is important to accurately model room acoustics. Thus, in this section we discuss underlying theory and existing techniques for sound scattering. We also discuss application of visibility techniques to accelerate existing methods for sound scattering.

Theory and Techniques

The geometric room acoustics can be generalized by an integral equation [54] called the *acoustic rendering equation* (see Eq. 1).

The acoustic rendering equation can be seen as an extension of the rendering equation in computer graphics [55].

$$L(x',\omega) = L_0(x',\omega) + \int_S R(x,x',\omega) L\left(x, \frac{x'-x}{|x'-x|}\right) dx \quad (1)$$

where L is final outgoing radiance, L_0 is emitted radiance and R is the *reflection kernel*, which describes how radiance at point x influences radiance at point x':

$$R(x,x',\omega) = \rho(x,x',\omega) G(x,x') V(x,x') P(x,x') \quad (2)$$

Here, ρ is the BRDF (Bidirectional Reflectance Distribution Function) of the surface at x, G is the form factor between x and x', V is the point-to-point visibility function, and P is a *propagation term* [54] that takes into account the effect of propagation delays. The latter is unique to sound rendering as visual rendering algorithms neglect propagation delays due to the high speed of light. Also, depending on the BRDF of a surface, different scattering properties of the surface, e.g. diffuse reflectance, can be modeled [56].

Several methods have been developed to solve the acoustic rendering equation. *Ray tracing* is a popular geometric algorithm for acoustic modeling [1] and can model specular and diffuse reflections easily. There has been much research in the computer graphics community to develop fast algorithms for ray tracing, by taking advantage of multi-core and many-core architectures, efficient scene hierarchies, and other acceleration techniques [57]. *Radiosity* is another another technique to model sound scattering [58, 59]. These algorithms operate by sampling the surface primitives and computing transfer operators which essentially encode the impulse response due to each sample at

every other sample. Radiosity based algorithms for sound scattering have been extended to efficiently handle moving sound sources and receiver [60].

Visibility Acceleration

Solving the acoustic rendering equation requires the computation of visibility between two points, $V(x,x')$. The visibility between two points ban be computed by shooting a ray from one point in the direction of the other. Scene hierarchies which organize the scene geometry can accelerate ray shooting and efficiently handle scenes with moving geometry [61]. Another possibility is to use from-region visibility data structures discussed in Section to to efficiently query visibility between two points. These visibility algorithms are compute and memory intensive for large scenes. However, for small scene used in room acoustics it is might be feasible to apply from-region visibility data structures to accelerate sound scattering computations.

ACKNOWLEDGMENT

This research is partially supported by Army Research Office, National Science Foundation, RDECOM, and Intel.

REFERENCES

1 A. Krokstad, S. Strom, and S. Sorsdal. Calculating the acoustical room response by the use of a ray tracing technique. *Journal of Sound and Vibration*, 8(1):118–125, July 1968.

2 Jont B. Allen and David A. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, April 1979.

3 Anish Chandak, Lakulish Antani, Micah Taylor, and Dinesh Manocha. Fastv: From-point visibility culling on complex models. *Computer Graphics Forum*, 28:1237–1246(10), 2009. doi: doi:10.1111/j.1467-8659.2009.01501.x.

4 Lakulish Antani, Anish Chandak, Micah Taylor, and Dinesh Manocha. Fast geometric sound propagation with finite edge diffraction. Technical Report TR10-011, University of North Carolina at Chapel Hill, 2010. `http://gamma.cs.unc.edu/BTM/FromRegion.pdf`.

5 M. A. Biot and I. Tolstoy. Formulation of wave propagation in infinite media by normal coordinates with an application to diffraction. In *Ocean Acoustics: Theory and Experiment in Underwater Sound*, pages 305–330. Acoustical Society of America, 1957. ISBN 088318527X.

6 Herman Medwin, Emily Childs, and Gary M. Jebsen. Impulse studies of double diffraction: A discrete huygens interpretation. *The Journal of the Acoustical Society of America*, 72(3):1005–1013, 1982. doi: 10.1121/1.388231.

7 F. Durand. *3D Visibility, Analysis and Applications*. PhD thesis, U. Joseph Fourier, 1999.

8 D. Cohen-Or, Y.L. Chrysanthou, C.T. Silva, and F. Durand. A survey of visibility for walkthrough applications. *Visualization and Computer Graphics, IEEE Transactions on*, 9(3):412–431, July-Sept. 2003. ISSN 1077-2626. doi: 10.1109/TVCG.2003.1207447.

9 T. Theoharis, G. Papaiannou, and E. Karabassi. The magic of the z-buffer: A survey. *Proc. of 9th International Conference on Computer Graphics, Visualization and Computer Vision, WSCG*, 2001.

10 J. Arvo and D. Kirk. A survey of ray tracing acceleration techniques. In *An Introduction to Ray Tracing*, pages 201–262, 1989.

11 J. Hasenfratz, M. Lapierre, N. Holzschuch, and F. Sillion. A survey of real-time soft shadows algorithms. In *Eurographics*. Eurographics, 2003. State-of-the-Art Report.

12 Brandon Lloyd, David Tuft, Sung-eui Yoon, and Dinesh Manocha. Warping and partitioning for low error shadow maps. In *Proceedings of the Eurographics Symposium on Rendering 2006*, pages 215–226. Eurographics Association, 2006.

13 Paul S. Heckbert and Pat Hanrahan. Beam tracing polygonal objects. In *Proc. of ACM SIGGRAPH*, pages 119–127, 1984. ISBN 0-89791-138-5.

14 R. Overbeck, R. Ramamoorthi, and W. R. Mark. A Real-time Beam Tracer with Application to Exact Soft Shadows. In *Eurographics Symposium on Rendering*, pages 85–98, Jun 2007.

15 Shaun Nirenstein. *Fast and Accurate Visibility Preprocessing*. PhD thesis, University of Cape Town, South Africa, 2003.

16 S. Nirenstein, E. Blake, and J. Gain. Exact from-region visibility culling. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, pages 191–202, Aire-la-Ville, Switzerland, Switzerland, 2002. Eurographics Association. ISBN 1-58113-534-3.

17 Ziv Gigus, John Canny, and Raimund Seidel. Efficiently computing and representing aspect graphs of polyhedral objects. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(6): 542–551, 1991. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/34.87341.

18 Frédo Durand, George Drettakis, and Claude Puech. The 3d visibility complex: a new approach to the problems of accurate visibility. In *Proceedings of the eurographics workshop on Rendering techniques '96*, pages 245–256, London, UK, 1996. Springer-Verlag. ISBN 3-211-82883-4.

19 Frédo Durand, George Drettakis, and Claude Puech. 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*, pages 89–100, New York, NY, USA, 1997. ACM Press/Addison-Wesley Publishing Co. ISBN 0-89791-896-7. doi: http://doi.acm.org/10.1145/258734.258785.

20 Xuetao Yin, Peter Wonka, and Anshuman Razdan. Generating 3d building models from architectural drawings: A survey. *IEEE Comput. Graph. Appl.*, 29(1):20–30, 2009. ISSN 0272-1716. doi: http://dx.doi.org/10.1109/MCG.2009.9.

21 J. Bittner and P. Wonka. Fast exact from-region visibility in urban scenes. *Eurographics Symposium on Rendering*, pages 223–230, 2005.

22 D. Luebke and C. Georges. Portals and mirrors: Simple, fast evaluation of potentially visible sets. In *ACM Interactive 3D Graphics Conference*, pages 105–108, Monterey, CA, 1995.

23 J. Bittner, V. Havran, and P. Slavik. Hierarchical visibility culling with occlusion trees. *Computer Graphics International, 1998. Proceedings*, pages 207–219, Jun 1998. doi: 10.1109/CGI.1998.694268.

24 Satyan Coorg and Seth Teller. Real-time occlusion culling for models with large occluders. In *SI3D '97: Proceedings of the 1997 symposium on Interactive 3D graphics*, pages 83–ff., New York, NY, USA, April 1997. ACM. ISBN 0-89791-884-3. doi: http://doi.acm.org/10.1145/253284.253312.

25 T. Hudson, D. Manocha, J. Cohen, M. Lin, K. Hoff, and H. Zhang. Accelerated occlusion culling using shadow frusta. In *Proc. of ACM Symposium on Computational Geometry*, pages 1–10, 1997.

26 Tomas Akenine-Möller and Timo Aila. Conservative and tiled rasterization using a modified triangle set-up. *journal of graphics tools*, 10(3):1–8, 2005.

27 F. Durand, G. Drettakis, J. Thollot, and C. Puech. Conservative visibility preprocessing using extended projections. *Proc. of ACM SIGGRAPH*, pages 239–248, 2000.

28 Jatin Chhugani, Budirijanto Purnomo, Shankar Krishnan,

Jonathan Cohen, Suresh Venkatasubramanian, David S. Johnson, and Subodh Kumar. vLOD: High-fidelity walkthrough of large virtual environments. *IEEE Transactions on Visualization and Computer Graphics*, 11(1):35–47, 2005. ISSN 1077-2626. doi: http://doi.ieeecomputersociety.org/10.1109/TVCG.2005.17.

29 S. J. Teller. *Visibility Computations in Densely Occluded Polyheral Environments*. PhD thesis, CS Division, UC Berkeley, 1992.

30 Seth J. Teller and Carlo H. Séquin. Visibility preprocessing for interactive walkthroughs. *SIGGRAPH Comput. Graph.*, 25(4):61–70, 1991. ISSN 0097-8930. doi: http://doi.acm.org/10.1145/127719.122725.

31 J. Foley, A. Van Dam, J. Hughes, and S. Feiner. *Computer Graphics: Principles and Practice*. Addison Wesley, Reading, Mass., 1990.

32 John D. Owens, David Luebke, Naga Govindaraju, Mark Harris, Jens Krüger, Aaron E. Lefohn, and Tim Purcell. A survey of general-purpose computation on graphics hardware. *Computer graphics forum*, 26(1):80–113, 2007.

33 Nvidia occlusion query. http://oss.sgi.com/projects/ogl-sample/registry/NV/occlusion_query.txt, 2002.

34 Peter Wonka, Michael Wimmer, Kaichi Zhou, Stefan Maierhofer, Gerd Hesina, and Alexander Reshetov. Guided visibility sampling. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers*, pages 494–502, New York, NY, USA, 2006. ACM. ISBN 1-59593-364-6. doi: http://doi.acm.org/10.1145/1179352.1141914.

35 Jiří Bittner, Oliver Mattausch, Peter Wonka, Vlastimil Havran, and Michael Wimmer. Adaptive global visibility sampling. In *SIGGRAPH '09: ACM SIGGRAPH 2009 papers*, pages 1–10, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-726-4. doi: http://doi.acm.org/10.1145/1576246.1531400.

36 Jont B. Allen and David A. Berkley. Image method for efficiently simulating small-room acoustics. *The Journal of the Acoustical Society of America*, 65(4):943–950, 1979. doi: 10.1121/1.382599.

37 Dirk Schröder and Tobias Lentz. Real-Time Processing of Image Sources Using Binary Space Partitioning. *Journal of the Audio Engineering Society (JAES)*, 54(7/8):604–619, July 2006.

38 James P. Chambers and Yves H. Berthelot. Time-domain experiments on the diffraction of sound by a step discontinuity. *The Journal of the Acoustical Society of America*, 96 (3):1887–1892, 1994. doi: 10.1121/1.410201.

39 Rendell R. Torres, U. Peter Svensson, and Mendel Kleiner. Computation of edge diffraction for more accurate room acoustics auralization. *The Journal of the Acoustical Society of America*, 109(2):600–610, 2001. doi: 10.1121/1.1340647.

40 Ville Pulkki, Tapio Lokki, and Lauri Savioja. Implementation and visualization of edge diffraction with image source method. In *Proceedings of the 112th AES Convention*, 2002.

41 Paul T. Calamia, U. Peter Svensson, and Thomas A. Funkhouser. Integration of edge diffraction calculations and geometrical-acoustics modeling. In *Proceedings of Forum Acusticum 2005*, 2005.

42 S. Laine, S. Siltanen, T. Lokki, and L. Savioja. Accelerated beam tracing algorithm. *Applied Acoustic*, 70(1):172–181, 2009.

43 Peter Svensson. Edge diffraction toolbox for matlab. http://www.iet.ntnu.no/ svensson/Matlab.html, 1999.

44 H. Lenhert. Systematic errors of the ray-tracing algoirthm. *Applied Acoustics*, 38:207–221, 1993.

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

46 Michael Vorlander. 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, 1989. doi: 10.1121/1.398336.

47 Christian Lauterbach, Anish Chandak, and Dinesh Manocha. Interactive sound rendering in complex and dynamic scenes using frustum tracing. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1672–1679, Nov.-Dec. 2007. ISSN 1077-2626. doi: 10.1109/TVCG.2007.70567.

48 Anish Chandak, Christian Lauterbach, Micah Taylor, Zhimin Ren, and Dinesh Manocha. AD-Frustum: Adaptive Frustum Tracing for Interactive Sound Propagation. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1707–1722, Nov.-Dec. 2008. ISSN 1077-2626. doi: 10.1109/TVCG.2008.111.

49 T. Funkhouser, N. Tsingos, and Jean-Marc Jot. Survey of Methods for Modeling Sound Propagation in Interactive Virtual Environment Systems. *Presence and Teleoperation*, 2003.

50 Micah Taylor, Anish Chandak, Qi Mo, Christian Lauterbach, Carl Schissler, and Dinesh Manocha. i-sound: Interactive gpu-based sound auralization in dynamic scenes. Technical Report TR10-006, University of North Carolina at Chapel Hill, 2010.

51 T. Funkhouser, N. Tsingos, I. Carlbom, G. Elko, M. Sondhi, J. West, G. Pingali, P. Min, and A. Ngan. A beam tracing method for interactive architectural acoustics. *Journal of the Acoustical Society of America*, 115(2):739–756, February 2004.

52 Thomas Funkhouser, Ingrid Carlbom, Gary Elko, Gopal Pingali, Mohan Sondhi, and Jim West. A beam tracing approach to acoustic modeling for interactive virtual environments. In *Proc. of ACM SIGGRAPH*, pages 21–32, 1998. ISBN 0-89791-999-8.

53 Bengt-Inge Dalenbäck, Mendel Kleiner, and Peter Svensson. A macroscopic view of diffuse reflection. *J. Audio Eng. Soc.*, 42:793–807, 1994.

54 Samuel Siltanen, Tapio Lokki, Sami Kiminki, and Lauri Savioja. The room acoustic rendering equation. *The Journal of the Acoustical Society of America*, 122(3):1624–1635, September 2007. doi: 10.1121/1.2766781.

55 James T. Kajiya. The rendering equation. In *Proc. of ACM SIGGRAPH*, pages 143–150, 1986. ISBN 0-89791-196-2.

56 Nicolas Tsingos, Carsten Dachsbacher, Sylvain Lefebvre, and Matteo Dellepiane. Instant sound scattering. In *Rendering Techniques (Proceedings of the Eurographics Symposium on Rendering)*, 2007.

57 Shirley et al. State of the art in interactive ray tracing. *SIGGRAPH Course Notes*, 2006.

58 Samuel Siltanen, Tapio Lokki, and Lauri Savioja. Frequency domain acoustic radiance transfer for real-time auralization. *Acta Acustica united with Acustica*, 95:106–117(12), 2009. doi: doi:10.3813/AAA.918132.

59 M. Bertram, E. Deines, J. Mohring, J. Jegorovs, and H. Hagen. Phonon tracing for auralization and visualization of sound. In *Proceedings of IEEE Visualization*, pages 151–158, 2005.

60 Lakulish Antani, Anish Chandak, Micah Taylor, and Dinesh Manocha. Direct-to-indirect acoustic radiance transfer. Technical Report TR10-012, University of North Carolina at Chapel Hill, 2010. `http://gamma.cs.unc.edu/Sound/diffuse/2010-tr2.pdf`.

61 I. Wald, W. Mark, J. Gunther, S. Boulos, T. Ize, W. Hunt, S. Parker, and P. Shirley. State of the art in ray tracing dynamic scenes. *Eurographics State of the Art Reports*, 2007.