3D Beam Tracing Based on Visibility Lookup for Interactive Acoustic Modeling

Dejan Marković*, *Member, IEEE*, Fabio Antonacci*, *Member, IEEE*, Augusto Sarti*, *Senior Member, IEEE*, and Stefano Tubaro*, *Senior Member, IEEE*

*Dip. di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Piazza Leonardo da Vinci 32, 20133 Milano, Italy.
E-mail: {dejan.markovic, fabio.antonacci, augusto.sarti, stefano.tubaro}@polimi.it.

Abstract—We present a method for accelerating the computation of specular reflections in complex 3D enclosures, based on acoustic beam tracing. Our method constructs the beam tree on the fly through an iterative lookup process of a precomputed data structure that collects the information on the exact mutual visibility among all reflectors in the environment (region-to-region visibility). This information is encoded in the form of visibility regions that are conveniently represented in the space of acoustic rays using the Plücker coordinates. During the beam tracing phase, the visibility of the environment from the source position (the beam tree) is evaluated by traversing the precomputed visibility data structure and testing the presence of beams inside the visibility regions. The Plücker parameterization simplifies this procedure and reduces its computational burden, as it turns out to be an iterative intersection of linear subspaces. Similarly, during the path determination phase, acoustic paths are found by testing their presence within the nodes of the beam tree data structure. The simulations show that, with an average computation time per beam in the order of a dozen of microseconds, the proposed method can compute a large number of beams at rates suitable for interactive applications with moving sources and receivers.

Index Terms—Acoustic simulation, room acoustics, beam tracing, visibility computation

1 Introduction

Room acoustics simulation has been a research topic of growing interest for quite a long time. Early works on room acoustics based on ray tracing date back over four decades [1]. Originally conceived for applications of architectural acoustics, geometric methods for modeling sound propagation are currently enjoying renewed popularity. They are among the most widespread methods in multimedia applications due to the their conceptual simplicity and computational efficiency [2]. A great deal of research effort has been devoted to interactive virtual acoustic environments [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. There are, however, numerous other application areas where such methods could play an important role. Environment-aware space-time audio processing methods [15] often use an acoustic simulation engine to swiftly model early reflections and perform room compensation [16]; exploit wall reflections in rendering applications [17]; use the energy of reflective paths to estimate the reflection coefficients of the walls of the environment [18]; etc.

Geometric methods are based on the concept of acoustic ray, which is an oriented line that (in a homogeneous medium) identifies a (locally) planar acoustic wavefront and is inherently perpendicular to it (i.e., it is collinear with the wave vector). As explained in [19], geometric methods can be categorized in stochastic and deterministic. Stochastic ray tracing [2] samples the space with rays in order to estimate the paths that link two points in a direct (line of sight) or indirect (wall-reflected, diffracted or diffused) fashion. Image source methods [20], [21], [22], on the other hand, compute the paths in a deterministic fashion but handle only specular reflections. Popular methods for computing diffuse reflections are based on acoustic radiosity [23], [24]. Today the most effective frameworks are based on hybrid methods [25], [26], [27], [28], which employ deterministic techniques for computing early reflections; and stochastic techniques for modeling more complex propagation phenomena such as diffusion.

In this manuscript we focus on specular reflections only. In particular, we are interested in a technique that is flexible enough to be applicable to environment-aware processing algorithms [17], [29], which require accurate knowledge of specular reflective paths, and, at the same time, is suitable for applications of spatial rendering of soundfields in a prescribed listening area [30], [31]. As a consequence, we do not just need an exact technique that models the *point-to-point* (source-receiver) acoustic response; we need an exact computation of all the components of the soundfield generated by the wall reflections. This can be done by computing not only image sources but also the compact bundles of rays (beams) that originate from them, which actually contribute to the resulting soundfield. In order to determine such beams (bundles of rays that contribute to the soundfield) we need to keep track of how they split and branch out as they interact with the reflectors of the environment. This is accomplished through beam tracing (BT), [32], [33], [34], which are deterministic techniques that overcome the exponentially increasing computing time of classical image source methods. Beam tracing methods, in fact, efficiently

Manuscript received 7 July 2015; revised 8 Dec. 2015; accepted 26 Dec. 2015. Date of publication 7 Jan. 2016.

compute all paths through an iterative lookup of the beam tree, a data structure that encodes the splitting/branching of beams [32]. Computing the beam tree, on the other hand, is a computationally demanding task.

To summarize: The set of direct and indirect paths that link source to receiver can be thought of as representing the *Point-to-Point* (P2P) visibility information. The beam tree, on the other hand, captures and encodes the *Point-to-Region* (P2R) visibility information. Traditional beam tracing methods efficiently compute P2P information through an iterative lookup of P2R information. Our goal is to be able to efficiently compute P2R information (beam tree), therefore it seems only natural to seek a method that does so through an iterative lookup of some *Region-to-Region* (R2R) visibility information [35]. This approach was originally proposed in [36] for planar (2D) geometries. In that work, R2R visibility was precomputed using the geometry of the reflectors only, and then encoded in the form of *visibility regions*. A simple geometric test was used for evaluating the presence of beams within such visibility regions and for iteratively recomputing the beam tree every time the source location changed.

In this manuscript we propose a generalization of [36] to the case of 3D geometries. This generalization is far from trivial because rays in 3D geometry are far more complex to represent in a parametric fashion than in the 2D case, therefore the concept of visibility regions turns out to exhibit tough challenges that need to be addressed with a certain progression. In order to make this manuscript as self-contained as possible we collected in section 2 all the background information that is needed in the following sections. After a brief overview of solutions proposed in the literature for efficiently computing the beam tree, we describe a particular type of homogeneous coordinates used in the literature to parameterize lines in 3D space, called Plücker coordinates, as well as methods used for visibility computations in the Plücker space. In section 3, which contains the innovative aspects of this contribution, we describe how to structure the visibility information for tracing purposes and present the algorithm that uses this structure for efficiently computing acoustic beams in arbitrary 3D environments. We finally discuss in detail the computational performance of our approach in section 4.

2 State of the Art and Background

2.1 State of the Art on Beam Tracing Methods

Several solutions have been proposed in the literature for efficiently computing the beam tree (P2R visibility). Binary Space Partitioning (BSP) [37] was used in [38] to split the geometric space into convex regions, so that a beam could be tested for incidence on the reflectors of the current region only. This method greatly reduces the number of reflectors to be tested at each iteration but the computational effort remains unevenly divided between beam tracing and path determination (PD). In order to further speed up the computation of the beam tree for walkthrough applications, [39] proposed three different improvements: priority-driven beam tracing, which uses psychoacoustics criteria to compute only the most perceptually relevant beams; bidirectional beam tracing, which limits the computational cost by combining two beam trees (originating from both source and receiver) of low reflection order; and the amortized beam tracing, partially inspired by the idea of conservative visibility determination [40], that encodes a superset of possible sequences of reflections from a region of space surrounding the source and performs the occlusion tests only for beams in the given superset. In [41] the path determination phase requires occlusion checks that are similar to those of the image source method, with the advantage that the process keeps the number of generated image sources under control. This acceleration technique enables interactive acoustic simulation in the presence of moving sources in simple environments. Frustum tracing [42], [43] casts a finite number of small beams (frusta) from the source and propagates them through the environment similarly to what rays do in ray tracing. Notice however, that in order to increase the accuracy of the method, frusta need to be split down into small-sized ones, which reduces the computational gain against traditional beam tracing. This enables a trade-off between accuracy and computational efficiency. Beam tracing methods can also be used for image rendering applications, for example to compute soft shadows from an area source [44].

P2R information is also used in FastV [45] in order to greatly speed up the computation of P2P information. In fact FastV, to our knowledge, is the fastest accurate geometric technique for computing specular reflections. More specifically, FastV uses frustum tracing to efficiently compute visibility and find image sources up to a prescribed order of reflection. It is worth noticing that the P2R information obtained by FastV is *conservative*, i.e., it contains objects that are not actually visible from the source location. The procedure for the computation of the P2P visibility discards the acoustic paths that are not feasible. Therefore, differently from the method proposed in this manuscript, although P2P is exactly computed by FastV [45] and amortized beam tracing in [39], the P2R information that it uses cannot be used for exactly determining beam-like contributions to the simulated sound field. An efficient and accurate computation of such contributions is crucial for applications such as rendering of virtual sound fields inside an extended listening area [30]. As a matter of fact, the P2R contains all the information that we need to structure a sound field as a superposition of individual beams, each originating from a different image-source.

For the reasons explained in the Introduction, we are interested in deterministic methods for a fast computation of the exact beam tree. In order to do so, instead of trading accuracy for computational efficiency, we perform most computationally-intensive tasks in an offline phase. Precomputation has been used in other methods for interactive auralization [46], [47], [48] as well. However, these methods are based on precomputed room impulse responses for sample source/receiver positions. Our method, on the other hand, does not require any type of sampling.

Finally, we remark that the methodology proposed in this manuscript explicitly models only specular reflective paths. In [49], [50] authors account for diffractive paths through the Uniform Theory of Diffraction (UTD). The UTD assumes that the diffracting edge is infinitely long and far from the source and receiver (high frequency approximation). An alternative approach, originally proposed in [51], is based on Biot-Tolstoy-Medwin (BTM) expression for

diffraction and makes no assumptions about frequency or geometry. However, BTM-based method is computationally expensive when compared to the UTD-based methods. The computational complexity of BTM-based method was addressed in [52]. The introduction of diffraction into beam tracing was investigated in [53], [54]. The advantage of using R2R visibility data structure for determining diffractive and diffusive acoustic paths has already been shown in [30] for two-dimensional environments. Modeling diffraction and diffusion, however, is beyond the scope of the current manuscript. However, the above techniques can be incorporated into the proposed framework in order to include the complex propagation phenomena.

2.2 Visibility in the Plücker Space

Extending the ideas proposed in [36] and [31] for 2D geometric space to 3D one is not a trivial task. In the 2D space we used a parameterization of rays (the *Ray Space*) for representing all geometric primitives. In the 3D space we would like to do the same. However, while the dimensionality of the geometric space grows of only 1, the number of degrees of freedom of the description of an oriented line (ray) goes from 2 to 4. In the 2D case a line is described by a single linear equation, whose parameters can be readily used for generating the ray space.

Unlike the 2D case, a line in 3D space is not a geometric object that splits the geometric space in two half-spaces (this is a property that, in 3D, is satisfied by planes instead). One could parameterize the line in 3D with a pair of constraints, i.e., two planes on which the line lies. This parameterisation, however, would not be unique, as any two planes picked from the pencil of planes passing through the line would be suitable for parameterizing that line. The parameterization of rays in 3D is a well-know problem [55]. Furthermore, working in a 3D geometric space brings quite a few additional challenges, particularly in the computation of visibility. For example, combined partial occlusions of a source on the part of planar reflectors delimited with rectilinear boundaries are known to cast a curved shadow [56]. In addition, in [57] the author shows that in the 3D geometry complex penumbra patterns could arise. Similar considerations are offered in [58] and [35].

2.2.1 The Ray Space in Plücker Coordinates

In order to overcome the inherent difficulties associated to the representation of rays in 3D, we chose to use the Plücker coordinates of lines [59]. This representation is fairly common in a wide literature of computer vision [60], and robotics [61] but has also been applied to problems of visibility computation. For example, such coordinates were used for finding lines that pass through four other lines [55]; stabbing convex polygons [62]; computing penumbra and antipenumbra of a light source [57]; etc.

Given two points $\boldsymbol{p}_A = [x_A, y_A, z_A]^T$ and $\boldsymbol{p}_B = [x_B, y_B, z_B]^T$, the Plücker coordinates of the two oriented lines (rays) passing through them are

$$\mathbf{l}_1 = k\begin{bmatrix}\boldsymbol{p}_B - \boldsymbol{p}_A \\ \boldsymbol{p}_A \times \boldsymbol{p}_B\end{bmatrix},\ \mathbf{l}_2 = h\begin{bmatrix}\boldsymbol{p}_A - \boldsymbol{p}_B \\ \boldsymbol{p}_B \times \boldsymbol{p}_A\end{bmatrix},\ k > 0,\ h > 0,$$

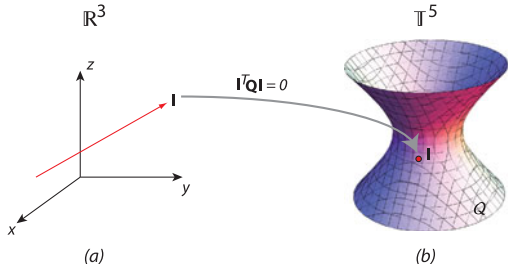


Fig. 1. A ray in the geometric space (a) and its representation in the ray space (b).

where "$\times$" is the vector cross product. The rays are therefore defined in a five-dimensional oriented projective space $\mathbb{T}^5$ [63]. Furthermore, from the above definition a ray is given by

$$\mathbf{l} = \begin{bmatrix}\mathbf{l}^d \\ \mathbf{l}^m\end{bmatrix},\ s.t.\ (\mathbf{l}^d)^T\mathbf{l}^m = 0, \tag{1}$$

where $\mathbf{l}^d$ is the vector given by the difference between point positions $\boldsymbol{p}_A$ and $\boldsymbol{p}_B$ (displacement or directional part) and $\mathbf{l}^m$ is the vector given by their vector product (moment or locational part). It is important to emphasize that not every point in $\mathbb{T}^5$ corresponds to a ray in $\mathbb{R}^3$. In fact, only points that satisfy the constraint $(\mathbf{l}^d)^T\mathbf{l}^m = 0$ can be retained as lines. All the points that satisfy the above constraint lie on the surface of a four-dimensional ruled quadric, known as the Plücker quadric, given by

$$Q = \{\mathbf{l} \in \mathbb{T}^5 \mid \mathbf{l}^T\mathbf{Q}\mathbf{l} = 0\},\ \ \mathbf{Q} = \begin{bmatrix}\mathbf{0} & \mathbf{I}_3 \\ \mathbf{I}_3 & \mathbf{0}\end{bmatrix}, \tag{2}$$

where $\mathbf{I}_3$ is 3×3 identity matrix. The Plücker quadric can be interpreted as the ray space, as every point on it corresponds to a ray in the 3D space. Quite clearly, we cannot visualize a four-dimensional quadric embedded in a five-dimensional projective space. In order to offer an intuitive understanding of the process, we will illustrate a two-dimensional ruled quadric embedded in a three-dimensional Euclidean space instead, as shown in Fig. 1.

Acoustic rays. As described above, we parametrize rays with six coordinates as in (1). These are homogeneous coordinates of a point in $\mathbb{T}^5$ lying on the quadric surface (2) (see Fig. 1).

When dealing with rays, it is also important to distinguish their orientations. For this purpose, Plücker coordinates work at our advantage. Indeed, along with the coordinates $\mathbf{l}_i$ of the ray passing through $\boldsymbol{p}_{Ai}$ and $\boldsymbol{p}_{Bi}$, we introduce a vector $\widetilde{\mathbf{l}}_i$ with permuted coordinates, i.e.

$$\widetilde{\mathbf{l}}_i = \begin{bmatrix}\mathbf{l}_i^m \\ \mathbf{l}_i^d\end{bmatrix} = \begin{bmatrix}\boldsymbol{p}_{Ai} \times \boldsymbol{p}_{Bi} \\ \boldsymbol{p}_{Bi} - \boldsymbol{p}_{Ai}\end{bmatrix}. \tag{3}$$

We interpret the coordinates of $\widetilde{\mathbf{l}}_i$ as the parameters of a hyperplane in $\mathbb{T}^5$, i.e. $\mathcal{I}_{\widetilde{\mathbf{l}}_i} = \{\mathbf{l} \in \mathbb{T}^5 | \widetilde{\mathbf{l}}_i^T\mathbf{l} = 0\}$. In order to understand how the representation in (3) allows us to infer the relative orientations of rays, consider the example shown in Fig. 2. We aim at testing the orientation of rays with respect to a given ray $\mathbf{l}_1$. Rays in Fig. 2a are oriented

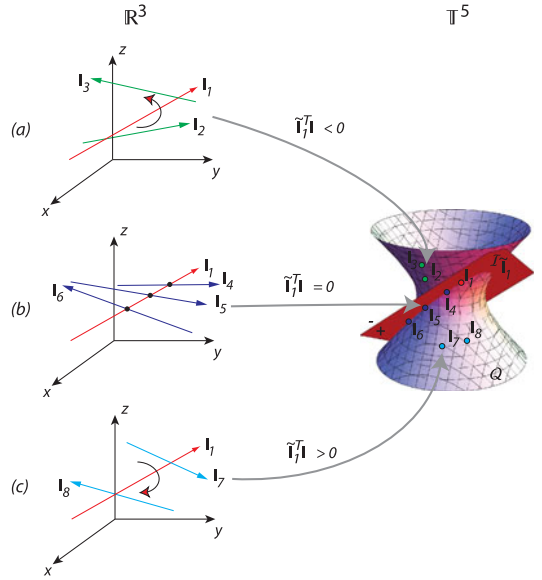


Fig. 2. Orientation of rays in the geometric space (a) and their configuration in the ray space (b).

counterclockwise around $\mathbf{l}_1$, whereas rays in Fig. 2c are oriented clockwise. Rays in Fig. 2b, finally, intersect $\mathbf{l}_1$. It is possible to discover the orientation of a ray $\mathbf{l}$ by testing the product $\tilde{\mathbf{l}}_1^T\mathbf{l}$. Indeed, counterclockwise (clockwise) rays are characterized by $\tilde{\mathbf{l}}_1^T\mathbf{l} > 0$ ($\tilde{\mathbf{l}}_1^T\mathbf{l} < 0$). Rays that lie exactly on $\tilde{\mathbf{l}}_1$ in $\mathbb{T}^5$ intersect (or are parallel, i.e. intersect at infinity) the ray $\mathbf{l}_1$ in $\mathbb{R}^3$. Noticing that $\tilde{\mathbf{l}} = \mathbf{Q}\mathbf{l}$, it is interesting to observe that the Plücker condition (2) can be rewritten as $\mathbf{l}^T\mathbf{Q}\mathbf{l} = \tilde{\mathbf{l}}^T\mathbf{l} = 0$. In other words, the Plücker condition states that a ray intersects itself.

Reflectors. In the geometric space reflectors are assumed to be planar surfaces. In the ray space we represent a reflector with the set of all the rays passing through it. More specifically, a reflector can be thought of as a combination of two oppositely oriented reflectors, each characterized by rays with different directions of incidence. To represent the oriented reflector we define a set of constraints on the relative position of rays with respect to its oriented edges, as shown in Fig. 3. The reflector $\mathcal{W}_i$ in 3D is uniquely identified by its oriented edges with Plücker coordinates $\mathbf{e}_i, i = 1, \ldots, N$ obtained as in (1) ($N = 3$ for the reflector of Fig. 3). In order for a ray to pass through the reflector, its orientation must be compatible with the orientation of all the edges forming the reflector, i.e. in the Plücker space $\mathbf{l}$ must lie in the N half-spaces determined by the edges $\mathbf{e}_i$. In the example in Fig. 3, the ray $\mathbf{l}$ must verify $\tilde{\mathbf{e}}_i^T\mathbf{l} \geq 0, i = 1, 2, 3$, where $\tilde{\mathbf{e}}_i = \mathbf{Q}\mathbf{e}_i$ are the parameters of the hyperplanes given by (3). Moreover, $\mathbf{l}$ must lie on the surface of the Plücker quadric. As a consequence, the oriented reflector in 3D space is given in $\mathbb{T}^5$ by the region on the Plücker quadric (this ensures that the points in $\mathbb{T}^5$ correspond to rays in $\mathbb{R}^3$) that lies in the N half-spaces determined by the oriented lines $\mathbf{e}_i$, $i = 1, \ldots, N$, that define the edges of the reflector, i.e.,

$$\mathcal{I}^{(+)}_{\mathbf{e}_i|_{i=1,\ldots,N}} = \{\mathbf{l} \in \mathbb{T}^5 \,|\, \tilde{\mathbf{e}}_i^T\mathbf{l} \rhd_i 0\,, \quad i = 1, \ldots, N \;\wedge\; \mathbf{l}^T\mathbf{Q}\mathbf{l} = 0\}, \tag{4}$$

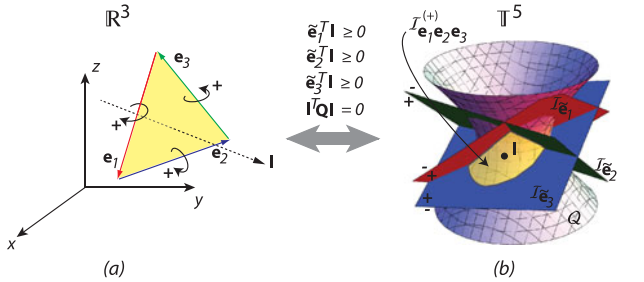


Fig. 3. Reflector in the geometric space (a) and its representation in the ray space (b).

where "$\rhd_i$" can represent either "$>$" or "$<$" depending on the orientation constraint with respect to the edge $\mathbf{e}_i$. The second oriented reflector $\mathcal{I}^{(-)}_{\mathbf{e}_i|_{i=1,\ldots,N}}$ has the opposite orientation with respect to the edges $\mathbf{e}_i$. The non-oriented reflector is finally given by the union of the two oriented reflectors, i.e.,

$$\mathcal{I}_{\mathbf{e}_i|_{i=1,\ldots,N}} = \mathcal{I}^{(+)}_{\mathbf{e}_i|_{i=1,\ldots,N}} \cup \mathcal{I}^{(-)}_{\mathbf{e}_i|_{i=1,\ldots,N}}.$$

2.2.2 *Visibility Information*

The visibility information will be used for speeding up the tracing of acoustic beams as they interact with the walls of the enclosure. The reflected beams are originated by an image (wall-mirrored) source. In accordance with the image source principle, when we evaluate the visibility of the environment from a mirrored source, we must not consider the reflectors in the half-space where the mirrored source lies. For this purpose, the definition of oriented reflectors turns out particularly useful. As a matter of fact, the visibility information that we strictly need for tracing acoustic beams is exactly the mutual visibility between all (oriented) reflectors in the environment. In the next paragraphs we define some primitives needed to approach the mutual visibility concept in a progressive fashion.

Region of Interest. All rays originated from one reflector $\mathcal{W}_i$ and falling onto another reflector $\mathcal{W}_j$ form the Region Of Interest (ROI) between the two reflectors, $\mathcal{R}(\mathbf{e}_i|_{i=1,\ldots,N}, \mathbf{e}_j|_{j=1,\ldots,M})$, where $\mathbf{e}_i|_{i=1,\ldots,N}$ and $\mathbf{e}_j|_{j=1,\ldots,M}$ are the edges of $\mathcal{W}_i$ and $\mathcal{W}_j$ respectively. More specifically, in the ray space, the ROI is given by the intersection of the corresponding oriented reflectors described by eq. (4) and depicted in Fig. 4.

Region of Visibility. If the environment has more than two reflectors, there can be mutual occlusions, which correspond to overlapping of ROIs in the ray space. We define the *Region of Visibility* (ROV) as the culling $\mathcal{R}^{(V)}(\mathbf{e}_i|_{i=1,\ldots,N}, \mathbf{e}_j|_{j=1,\ldots,M})$ of a reflector, i.e., is the portion of the ROI in

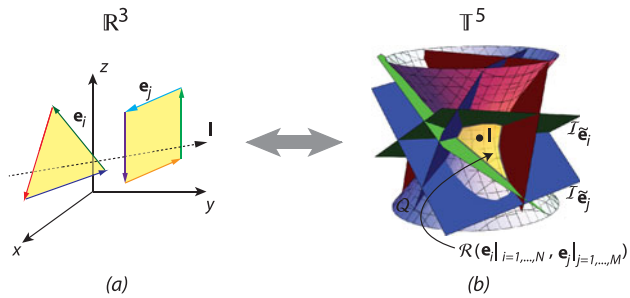


Fig. 4. Visibility region in the geometric space (a) and its representation in the ray space (b).

which the occluded portions of $\mathcal{R}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M})$ have been removed. Notice that the construction of a ROV from the overlapping of multiple ROIs requires the inclusion of new constraints, the removal of redundant ones and, possibly, the splitting of the region (e.g., if an occluder divides the ROI into two ROVs). Furthermore, while evaluating the mutual visibility between reflectors in the environment, only the intersections on the surface of the Plücker quadric must be taken into account, i.e., only the events that correspond to visibility changes in the 3D geometric space. As a consequence, unlike the 2D case, the removal of occluded portions of visibility regions is no longer a linear program, and a new, dedicated, algorithm for detecting occlusions, adding/removing constraints and splitting regions needs to be adopted. For comprehensive surveys of visibility computation algorithms, the interested reader can refer, for example, to [64] and [35]. For visibility culling algorithms in the space of Plücker coordinates we refer to the approaches described in [65], [66], [67], and [68]. Our implementation, in particular, is based on [68].

3 Tracing Beams Through Visibility Lookup

The previous section provided us with the description of acoustic primitives in the Ray Space. The important fact is that the precomputation of the ROVs relieves the algorithm of the burden of having to handle the occlusion among reflectors during the tracing phase. In fact, as the mutual visibility between reflectors depends on environment geometry only, all occlusions are handled in this pre-processing stage. This complexity reduction becomes relevant in occluded environments, where only a limited number of reflectors are mutually visible. The actual tracing of beams can therefore be performed "on the fly", as a lookup on a precomputed data structure that organizes the information of mutual visibility among reflectors. Each step of this procedure is described in detail in the following paragraphs.

3.1 Visibility Diagrams

In order to organize the information appropriately for the tracing of beams, ROVs are organized in the form of "visibility diagrams". In particular, the visibility diagram $\mathcal{D}(\mathbf{e}_i|_{i=1,\dots,N})$ of a reflector $\mathcal{W}_i$ is the collection of ROVs from that reflector, i.e.,

$$\begin{aligned}\mathcal{D}(\mathbf{e}_i|_{i=1,\dots,N}) = \{\mathcal{R}^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M}) \;\mid\; \\ \forall\, \mathcal{I}_{\mathbf{e}_j|_{j=1,\dots,M}} \neq \mathcal{I}_{\mathbf{e}_i|_{i=1,\dots,N}} \;\wedge \\ \mathcal{R}^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M}) \neq \emptyset\},\end{aligned}$$

where $\mathcal{I}_{\mathbf{e}_j|_{j=1,\dots,M}}$ are ray space representations of all the reflectors $\mathcal{W}_j$ that are visible from the reflector $\mathcal{W}_i$.

An example of visibility diagram is depicted in Fig. 5. The reflector $\mathcal{W}_i$, defined by edges $\mathbf{e}_i|_{i=1,\dots,N}$, observes three reflectors:

- reflector $\mathcal{W}_j$, defined by edges $\mathbf{e}_j|_{j=1,\dots,M}$, which is completely visible. As a consequence, the ROV $\mathcal{R}^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M})$ is the same as the ROI $\mathcal{R}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M})$;
- reflector $\mathcal{W}_k$, defined by edges $\mathbf{e}_k|_{k=1,\dots,K}$, which is partially occluded by $\mathcal{W}_j$. Thus, after culling the ROI $\mathcal{R}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_k|_{k=1,\dots,K})$ generates the two ROVs $\mathcal{R}_1^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_k|_{k=1,\dots,K})$ and $\mathcal{R}_2^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_k|_{k=1,\dots,K})$. The two regions are determined by the constraints $\mathbf{e}_i$ and $\mathbf{e}_k$, which ensure that the rays that depart from $\mathcal{I}_{\mathbf{e}_i|_{i=1,\dots,N}}$ are incident on $\mathcal{I}_{\mathbf{e}_k|_{k=1,\dots,K}}$, and by a subset of constraints $\mathbf{e}_j$, which ensure that the rays do not intersect $\mathcal{I}_{\mathbf{e}_j|_{j=1,\dots,M}}$;
- reflector $\mathcal{W}_l$, defined by edges $\mathbf{e}_l|_{l=1,\dots,L}$, which is completely occluded by $\mathcal{W}_k$ and whose ROV is an empty set.

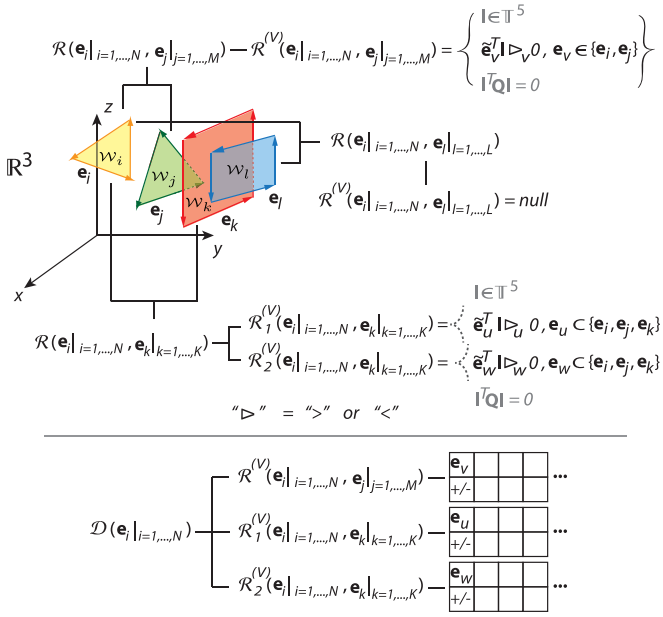


Fig. 5. An example of visibility diagram.

Therefore, the visibility diagram of the reflector $\mathcal{W}_i$ from Fig. 5 is made by three visibility regions. For each ROV the diagram contains the list of active constraints and the corresponding signs. The pseudocode for the construction of the visibility diagrams is shown in Algorithm 1.

3.2 Representation of Point-Like Sources and Receivers

In order to perform beam tracing using the visibility information encoded with the Plücker coordinates, we have to represent acoustic sources (e.g., loudspeakers) and receivers (e.g., microphones) in the Plücker space. Acoustic sources and receivers can be seen as points in the geometric space. A point is identified in the ray space by the parameters of all the rays that pass through it. However, given a point of Cartesian coordinates $\boldsymbol{p}_P = [x_P, y_P, z_P]^T$, it is not so straightforward to force a generic line $\mathbf{l}$, represented using Plücker coordinates, to pass trough $\boldsymbol{p}_P$, as it is in the 2D case [31]. The point $\boldsymbol{p}_P$ and the line $\mathbf{l}$ are defined in two different spaces, and therefore they can't "speak" to each other directly. However, we can constrain a ray $\mathbf{l}$ to pass through $\boldsymbol{p}_P$ by constraining it to intersect three non-coplanar lines, $\mathbf{l}_1\mathbf{l}_2$, $\mathbf{l}_3$, all passing through the point $\boldsymbol{p}_P$, i.e.,

$$\tilde{\mathbf{l}}_1^T\mathbf{l} = 0, \quad \tilde{\mathbf{l}}_2^T\mathbf{l} = 0, \quad \tilde{\mathbf{l}}_3^T\mathbf{l} = 0.$$

As a consequence the ray $\mathbf{l}$ belongs to the null space of $\mathbf{L} = [\tilde{\mathbf{l}}_1, \tilde{\mathbf{l}}_2, \tilde{\mathbf{l}}_3]^T$. We choose $\tilde{\mathbf{l}}_1, \tilde{\mathbf{l}}_2, \tilde{\mathbf{l}}_3$ to be three perpendicular

lines, each parallel to one of the main axes (x, y and z). Therefore, we can write that

$$\mathbf{L} = \begin{bmatrix} 0 & z_P & -y_P & 1 & 0 & 0 \\ -z_P & 0 & x_P & 0 & 1 & 0 \\ y_P & -x_P & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5)$$

If we assume that $x_P, y_P, z_P \neq 0$, a basis for the null space of $\mathbf{L}$ is given by

$$\mathbf{U}_P = \begin{bmatrix} 0 & 1/z_P & x_P/z_P \\ 1/x_P & y_P/(x_P z_P) & y_P/z_P \\ 0 & 0 & 1 \\ -z_P/x_P & -y_P/x_P & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}. \quad (6)$$

An equivalent representation, not shown for reasons of space, can be found also when one or more of the above conditions on x_P, y_P, z_P are not met.

Algorithm 1. Pseudocode for the Construction of the Visibility Diagrams

function *Visibility evaluation* ($\mathcal{I}_{\mathbf{e}_i|_{i=1,\dots,N}}$)
  initialize *the visibility diagram* $\mathcal{D}(\mathbf{e}_i|_{i=1,\dots,N}) = \emptyset$
  for *all reflectors* $\mathcal{I}_{\mathbf{e}_j|_{j=1,\dots,M}} \neq \mathcal{I}_{\mathbf{e}_i|_{i=1,\dots,N}}$ **do**
    initialize *the ROV* $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j) = \emptyset$[a]
    compute *the ROI* $\mathcal{R}(\mathbf{e}_i, \mathbf{e}_j)$
    add $\mathcal{R}(\mathbf{e}_i, \mathbf{e}_j)$ *to* $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$
    for *all possible occluders*[b] $\mathcal{I}_{\mathbf{e}_k|_{k=1,\dots,K}}$ **do**
      compute *the ROI* $\mathcal{R}(\mathbf{e}_i, \mathbf{e}_k)$
      for $\forall \mathcal{R}_r^{(V)}(\mathbf{e}_i, \mathbf{e}_j) \in \mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$[c] **do**
        test *intersection* $\mathcal{R}_r^{(V)}(\mathbf{e}_i, \mathbf{e}_j) \cap \mathcal{R}(\mathbf{e}_i, \mathbf{e}_k)$
        if *intersection = true* **then**
          cull $\mathcal{R}_r^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ *into* $\mathcal{R}_o^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$[d]
          remove $\mathcal{R}_r^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ *from* $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$
          add $\mathcal{R}_o^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ *to* $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$
        end
      end
    end
    add $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ *to* $\mathcal{D}(\mathbf{e}_i|_{i=1,\dots,N})$
  end
  return $\mathcal{D}(\mathbf{e}_i|_{i=1,\dots,N})$
end

[a] abbreviated for $\mathcal{R}^{(V)}(\mathbf{e}_i|_{i=1,\dots,N}, \mathbf{e}_j|_{j=1,\dots,M})$;
[b] the occluders can be sorted by importance by shooting test rays and measuring, for each occluder, the number of rays intersecting it, as suggested in [66];
[c] in a general scenario $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ is composed by multiple connected regions (see Fig. 5 for an example); $\mathcal{R}_r^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$ denotes the rth connected region that composes $\mathcal{R}^{(V)}(\mathbf{e}_i, \mathbf{e}_j)$;
[d] empty set in case of complete occlusion.

A ray $\mathbf{l}$ belongs to the vector space spanned by vectors $\mathbf{U}_P = [\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3]$, if it can be written as a linear combination of $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$:

$$\mathbf{l} = \mathbf{u}_1 a_1 + \mathbf{u}_2 a_2 + \mathbf{u}_3 a_3 = \mathbf{U}_P \mathbf{a}, \quad (7)$$

where $\mathbf{a} = [a_1, a_2, a_3]^T$. From (6) and (7) an important result follows:

Theorem 3.1. *In the ray space the point $\boldsymbol{p}_P$ (acoustic source or receiver) is given by the set of all rays that pass through it, i.e. by the plane (spanned by (6)) that lies completely on the surface of the (ruled) Plücker quadric embedded in $\mathbb{T}^5$, i.e.,*

$$\mathcal{I}_{\boldsymbol{p}_P} = \{\mathbf{l} \in \mathbb{T}^5 | \mathbf{l} = \mathbf{U}_P \mathbf{a}, \mathbf{a} \in \mathbb{T}^2\}. \quad (8)$$

Proof. Notice that $\mathbf{U}_P(k\mathbf{a}) = k(\mathbf{U}_P\mathbf{a}) = k\mathbf{l}$, i.e., the coefficients $\mathbf{a}$ are homogeneous coordinates that specify a point on the projective plane $\mathbb{T}^2$. Furthermore, from (6) it is not difficult to verify that

$$\begin{aligned} &\mathbf{u}_i^T \mathbf{Q} \mathbf{u}_j = 0, \ i, j = 1, 2, 3, \\ &\quad \Longrightarrow \mathbf{l}^T \mathbf{Q} \mathbf{l} = \sum_{i=1}^{3} \sum_{j=1}^{3} a_i a_j \mathbf{u}_i^T \mathbf{Q} \mathbf{u}_j = 0, \end{aligned} \quad (9)$$

or, equivalently, each ray $\mathbf{l}$ given by (7) inherently satisfy the Plücker constraint (2). □

As we have seen, going from a 2D geometric space to a 3D one, visual events become nonlinear and non-convex. The ray space becomes a non-convex quadric ruled surface embedded in $\mathbb{T}^5$ (four degrees of freedom). However, the Theorem 3.1 has important consequences for the beam tracing phase:

Corollary 3.1.1. *Going from 2D to 3D the dimensional growth of the point representation is exactly one. In fact, all rays passing through a point in 2D have 1 degree of freedom, i.e. in the ray space they belong to a projective line $\mathbb{T}^1$ [31]; in 3D the rays passing through a point have 2 degrees-of-freedom, i.e. in the ray space they belong to the projective plane $\mathbb{T}^2$.*

Corollary 3.1.2. *As all the rays passing through the point lie completely on the surface of the ruled quadric, no intersection is needed. Indeed, the visibility events from a point are, once again, linear, as observed in [57], [56], [69]. We will see that the beam tracing is again a linear procedure, which consists of intersecting the reflected beam with the precomputed visibility regions of the source reflector.*

As a consequence, even in 3D, the beam tracing procedure has a reasonable computational burden that allows efficient tracing of acoustic beams.

3.3 Representation of a Beam

The last geometric primitive we have to define in order to perform visibility-based beam tracing is the acoustic beam. We recall that a beam is a compact bundle of rays that originate from an acoustic source and fall onto a single reflector. The beam is defined by: its source (real or image); the reflector that originates the reflection (except for the direct beams); the reflector it falls onto; and the edges $\mathbf{e}_i$ that delimit the beam. In the ray space a beam is represented by a portion of the projective plane (8) representing the source, delimited by a number of hyperplanes that correspond to the edges $\mathbf{e}_i$ of the beam. Notice that in going from a 2D space to a 3D space, the dimensionality of beams grows of only one degree of freedom (in 2D a beam was a portion of a projective line [31]) despite the fact that the ray space expands from $\mathbb{T}^2$ [31] to $\mathbb{T}^5$.

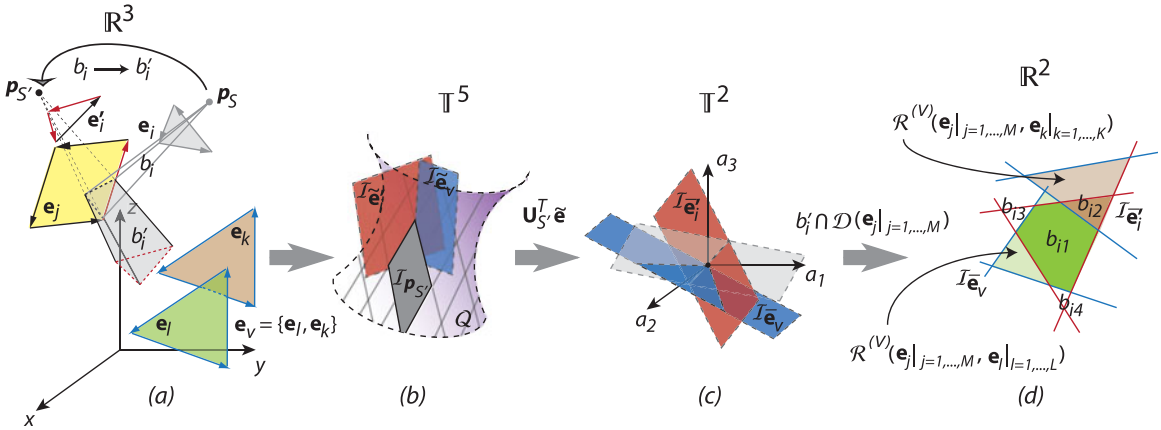


Fig. 6. Steps of the visibility-based beam tracing algorithm: 1) Compute the reflected bundle of rays (a); 2) Perform the dimensionality reduction (b), (c); 3) Split the reflected bundle of rays into new beams (d).

In order to determine the edges $\mathbf{e}_i$ that delimit the beam, we project the constraints $\tilde{\mathbf{e}}_i$ from $\mathbb{T}^5$ onto the projective plane $\mathbb{T}^2$, the latter being defined by the coefficients $\mathbf{a}$ of all the lines that pass through the source (see eqs. (6) and (7)). The projection operator

$$\begin{aligned} g_P : \mathbb{T}^5 &\rightarrow \mathbb{T}^2 \\ \tilde{\mathbf{e}}_i \mapsto \bar{\mathbf{e}}_i &= g_P(\tilde{\mathbf{e}}_i) = \mathbf{U}_P^T \tilde{\mathbf{e}}_i \ , \end{aligned} \tag{10}$$

$$\mathcal{I}_{\tilde{\mathbf{e}}_i} \mapsto \mathcal{I}_{\bar{\mathbf{e}}_i} = \{\mathbf{a} \in \mathbb{T}^2 | \bar{\mathbf{e}}_i{}^T \mathbf{a} = 0\} \ , \tag{11}$$

is implemented by the matrix $\mathbf{U}_P$ in eq. (6). Using the coefficients $\mathbf{a}$ and the projections $\bar{\mathbf{e}}_i$ we can still test the orientation of the ray $\mathbf{l}$ with respect to the edge $\mathbf{e}_i$, as $\tilde{\mathbf{e}}_i^T \mathbf{l} = \tilde{\mathbf{e}}_i^T \mathbf{U}_P \mathbf{a} = (\mathbf{U}_P^T \tilde{\mathbf{e}}_i)^T \mathbf{a} = \bar{\mathbf{e}}_i^T \mathbf{a}$. As a consequence, exploiting this dimensionality reduction, during the beam tracing phase we can perform geometric tests (i.e., find intersections of reflected beams with visibility regions) in $\mathbb{T}^2$ instead of $\mathbb{T}^5$, with a clear advantage on the computational complexity.

3.4 Beam Tracing

The branching of the beam tree (splitting of reflected bundles of rays into beams) is entirely controlled by the visibility conditions from the viewpoint of the source. With reference to Fig. 6, we show how we use the visibility diagrams, which encode the mutual visibility between reflectors, to iteratively trace beams. For each beam that falls onto a reflector we do the following:

1) Compute the reflected bundle of rays. With reference to Fig. 6a, let us consider the reflection of the beam b_i, with (image) source $\boldsymbol{p}_S$ and edge constraints $\tilde{\mathbf{e}}_i$, $i = 1, \dots, N$, from the reflector $\mathcal{W}_j$. First the source $\boldsymbol{p}_S$ and edges $\tilde{\mathbf{e}}_i$ are reflected with respect to the plane that the reflector lies on. The result is the reflected bundle of rays b'_i with image source $\boldsymbol{p}_{S'}$ and constraints $\tilde{\mathbf{e}}'_i$, $i = 1, \dots, N$. Let π be the 4×1 vector containing the plane coefficients, the reflection is performed using the reflection matrix [60]

$$\mathbf{M}_\pi = \mathbf{I}_4 - 2\mathbf{G}\frac{\pi\pi^T}{\pi^T \mathbf{G} \pi}, \ \mathbf{G} = \begin{bmatrix} \mathbf{I}_3 & 0 \\ 0 & 0 \end{bmatrix}. \tag{12}$$

Notice that the reflection matrix $\mathbf{M}_\pi$ is an isometry. Therefore we can write that

$$\mathbf{M}_\pi = \begin{bmatrix} \mathbf{N} & \mathbf{n} \\ \mathbf{0} & 1 \end{bmatrix}.$$

Using the homogeneous coordinates we have $\mathbf{p}_{S'} = \mathbf{M}_\pi \mathbf{p}_S$, therefore the Cartesian coordinates of the image source $\boldsymbol{p}_{S'}$ are obtained as $\boldsymbol{p}_{S'} = \mathbf{N}\boldsymbol{p}_S + \mathbf{n}$. Given a generic edge $\mathbf{e}$ passing through two points $\boldsymbol{p}_{P_1}$ and $\boldsymbol{p}_{P_2}$, the hyperplane in the ray space associated to it is given by

$$\tilde{\mathbf{e}} = \begin{bmatrix} \boldsymbol{p}_{P_1} \times \boldsymbol{p}_{P_2} \\ \boldsymbol{p}_{P_2} - \boldsymbol{p}_{P_1} \end{bmatrix} = \begin{bmatrix} \mathbf{e}^m \\ \mathbf{e}^d \end{bmatrix}.$$

The hyperplane $\tilde{\mathbf{e}}'$ in the ray space associated to the edge $\mathbf{e}'$ is obtained by mirroring $\tilde{\mathbf{e}}$ against π, and is given by

$$\begin{aligned} \tilde{\mathbf{e}}' &= \begin{bmatrix} (\mathbf{N}\boldsymbol{p}_{P_1} + \mathbf{n}) \times (\mathbf{N}\boldsymbol{p}_{P_2} + \mathbf{n}) \\ (\mathbf{N}\boldsymbol{p}_{P_2} + \mathbf{n}) - (\mathbf{N}\boldsymbol{p}_{P_1} + \mathbf{n}) \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{N}\boldsymbol{p}_{P_1} \times \mathbf{N}\boldsymbol{p}_{P_2} + \mathbf{n} \times \mathbf{N}(\boldsymbol{p}_{P_2} - \boldsymbol{p}_{P_1}) + \mathbf{n} \times \mathbf{n} \\ \mathbf{N}(\boldsymbol{p}_{P_2} - \boldsymbol{p}_{P_1}) \end{bmatrix} \\ &= \begin{bmatrix} \det(\mathbf{N})(\mathbf{N}^{-1})^T \mathbf{e}^m + \mathbf{n} \times (\mathbf{N}\mathbf{e}^d) + \mathbf{n} \times \mathbf{n} \\ \mathbf{N}\mathbf{e}^d \end{bmatrix}. \end{aligned} \tag{13}$$

It is important to notice that, from Theorem 3.1, in the ray space the representation of the bundle of rays b'_i *lies completely on the surface of the Plücker quadric*, bounded by the hyperplanes $\tilde{\mathbf{e}}'_i$, $i = 1, \dots, N$, as depicted in Fig. 6b;

2) Perform the dimensionality reduction. In order to perform the beam subdivision there is no need to remain in a six-dimensional space and deal with a non-convex quadric surface. Indeed, since all rays of the bundle are bound to pass through the image source location $\boldsymbol{p}_{S'}$, they are guaranteed to lie on the Plücker quadric, as from Theorem 3.1. To perform the visibility reduction, the hyperplanes $\tilde{\mathbf{e}}_v$, $v = 1, \dots, W$, involved in the definition of the visibility diagram of the reflector $\mathcal{W}_j$ are now projected on $\mathbb{T}^2$ through the operator defined in (11) for the point $\boldsymbol{p}_{S'}$. Similarly, we project also the hyperplanes $\tilde{\mathbf{e}}'_i$ associated to the edges of the beam. The result is a collection of linear constraints, $\bar{\mathbf{e}}'_i$ and $\bar{\mathbf{e}}_v$, within a two-dimensional projective space $\mathbb{T}^2$ as shown in Fig. 6 (c). The coordinates that span the projective space $\mathbb{T}^2$ are given by $\mathbf{a} = [a_1, a_2, a_3]^T$;

3) Split the reflected bundle of rays into new beams. The reflected bundle of rays b'_i is then split into new beams that intersect the visibility diagram $\mathcal{D}(\mathbf{e}_j|_{j=1,\dots,M})$ of the reflector $\mathcal{W}_j$. This procedure is performed in the projective space $\mathbb{T}^2$ spanned by the coordinates $\mathbf{a}$. For clarity of visualization, in Fig. 6d we depict the splitting process in a two-dimensional Euclidean space obtained intersecting $\mathbb{T}^2$ with a prescribed plane. A simple example of the visibility diagram $\mathcal{D}(\mathbf{e}_j|_{j=1,\dots,M})$ with two computed ROVs, $\mathcal{R}^{(V)}(\mathbf{e}_j|_{j=1,\dots,M}, \mathbf{e}_k|_{k=1,\dots,K})$ and $\mathcal{R}^{(V)}(\mathbf{e}_j|_{j=1,\dots,M}, \mathbf{e}_l|_{l=1,\dots,L})$ is shown in Fig. 6d. The overlap between the regions has been managed during the visibility pre-computation phase and the visibility diagram shows unoccluded convex portions defined by a number of linear constraints $\bar{\mathbf{e}}_v$. Finally, the new beams are computed by intersecting the reflected bundle of rays b'_i, specified by the constraints $\bar{\mathbf{e}}'_i$, with the regions of the visibility diagram $\mathcal{D}(\mathbf{e}_j|_{j=1,\dots,M})$, as shown in Fig. 6d. Some beams encounter reflectors along their path (b_{i1} and b_{i2}) and originate new reflections, others keep propagating to infinity (b_{i3} and b_{i4});

4) Add new branches to the beam tree. The beams $b_{i1}, b_{i2}, b_{i3}, b_{i4}$ originated by the reflection of b_i onto the reflector $\mathcal{W}_j$ are added to the beam tree as branches of the node associated to b_i;

5) Repeat the procedure for each beam that falls onto a reflector. The recursive procedure stops when the preassigned order of reflection is reached or when the beams "die out" (i.e., when their magnitude drops below a preassigned threshold).

The pseudocode for the construction of the beam tree is shown in Algorithm 2.

3.5 Path Determination

Finally, the acoustic paths that link the acoustic source to the receiver are found testing the presence of the source inside each beam of the beam tree data structure. Therefore, the path determination is essentially a recursive tree traversal algorithm that performs the following, simple, geometric test for each node (beam) of the beam tree. Let $\boldsymbol{p}_S$ be the image source of the beam, $\boldsymbol{p}_R$ the receiver, π_S the plane associated with the reflector that originates the beam, π_R the plane of the reflector that the beam falls onto, and $\mathbf{e}_i$, $i = 1, \dots, N$, the edges that delimit the beam. The receiver is inside the beam if all the following conditions are meet:

$$\begin{aligned} &\pi_S^T \boldsymbol{p}_R \rhd_s 0, \\ &\pi_R^T \boldsymbol{p}_R \rhd_r 0, \\ &\widetilde{\mathbf{e}}_i^T \mathbf{l}_{SR} \rhd_i 0, \quad i = 1, \dots, N, \end{aligned}$$

where $\mathbf{l}_{SR}$ is the ray going from $\boldsymbol{p}_S$ to $\boldsymbol{p}_R$, and $\rhd$ can be either $<$ or $>$ depending on the required position with respect to planes π_S, π_R and edges $\mathbf{e}_i$.

4 Performance Evaluation

4.1 System Implementation

In order to analyse the complexity of the proposed approach, a first prototype has been developed in C++ using Ogre3D (Object-oriented Graphics Rendering Engine) library for the 3D visualization and Armadillo library for the linear algebra calculations required by the tracing algorithm. All tests are performed using a single thread running on a Intel Core i7-4870HQ CPU processor at 2.50 GHz.

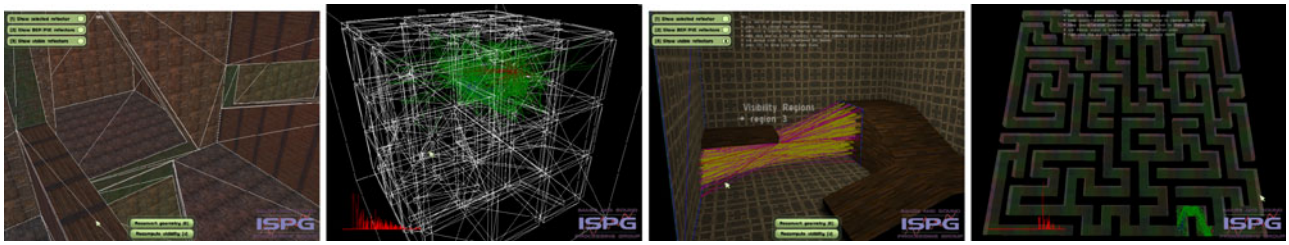
Algorithm 2. Pseudocode for the Visibility-Based Beam Tracing Algorithm

```
function traceBeams(b_i)
  if nextBeams b_ij not computed then
    if refOrder < maxOrder then
      b'_i = reflectBeam (b_i)[a]
      b_ij = subdivideBeam(b'_i)[b]
    else
      return
    end
  end
  if refOrder < maxOrder then
    for ∀b_ij do
      traceBeams (b_ij)
    end
  end
end

function computeBeamTree()
  b_0 = computeFirstBeams(p_S)[c]
  traceBeams(b_0)
end

function computeFirstBeams(p_S)
  Bn = getBSPnode(p_S)
  PVS = getPotentiallyVisibleSet(Bn)
  b_0 = initializeBeamTree(p_S)
  for ∀W_jϵ PVS do
    if isVisible(p_S,W_j) then
      b_0j = initializeBeam (b_0, W_j)
      for ∀W_kϵ PVS, W_k ≠ W_j do
        if occlusionDetected (b_0j,W_k) then
          b_0j = occlusionCulling (b_0j,W_k)
        end
      end
    end
  end
end
```

[a] see (12) and (13);
[b] the intersection $b'_i \cap \mathcal{D}(\mathbf{e}_j|_{j=1,\dots,M})$ is performed after the dimensionality reduction (11);
[c] the first (direct) beams do not have a reference reflector and therefore can not be computed using the visibility regions; a traditional beam tracing algorithm is used for their computation.

4.2 Environment Complexity

In order to assess the effectiveness of our algorithm, we analyze its behavior for different types of environments. One straightforward way to classify environments is based on their size, i.e., the total number of triangles that compose them. We call this parameter the *geometric complexity T* of the environment. However, intuitively, the raw number of triangles is not enough to describe the complexity of an environment. For what concerns the modeling of the visibility, in order to classify the complexity of environments it is more important to assess how reflectors see each other, i.e., the *visual complexity R* of the environment. In particular, we measure the visual complexity as the average number of visibility regions that are visible from a single reflector.



(a) cube3A interior (b) cube3A exterior (c) room (d) maze

Fig. 7. Environment geometries used in simulations.

Fig. 8 shows the comparison between the 9 different environments considered in this work, based on their geometric and visual complexities. The test environments, more specifically, are

- *cube1*: A cubic room, the simplest possible environment included as a basis for comparison with other environments;
- *cube2*, *cube3A*, *cube4*, *cube5*, *cube6*: This class of environments is made by a variable number of cubic rooms placed on $n \times n \times n$ grid connected by openings, where $n = 2, 3, 4, 5, 6$. These environments were generated by an algorithm that picks shape and location of the openings at random, while keeping the visual complexity approximatively constant. An example of the environment interiors is shown in Fig. 7a, while Fig. 7b shows a wireframe external view of the same environment
- *cube3B*: Similarly to cube3A it is a $3 \times 3 \times 3$ room grid. However, the number of portals is much higher and, as a consequence, it has a higher visual complexity. On the other hand, the geometric complexity is only modestly increased;
- *room*: This is an environment with a low geometric complexity and high visual complexity. It is a small room, but the presence of objects yields many occlusions, and therefore ROIs are splitted into multiple regions. Fig. 7c shows the environment and an example of visibility region. Pink rays represent the rays that determine the border of the visibility region, while yellow rays lie inside the region;

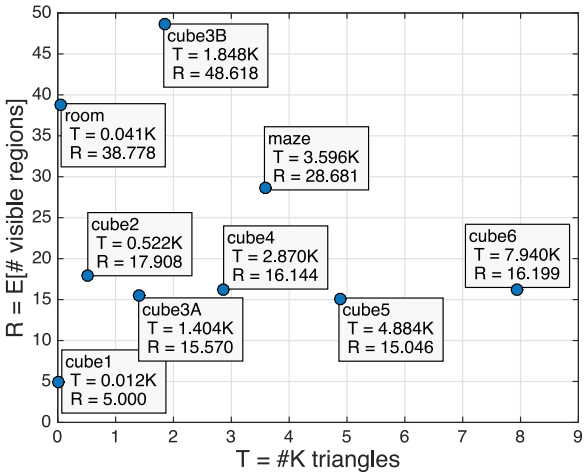


Fig. 8. Geometric and visual complexity of the test environments.

- *maze*: This is a maze-like environment, shown in Fig. 7d. This environment is characterized by a balanced geometric/visual complexity.

Although computationally expensive, the visibility evaluation only requires the information on the geometry of the environment and can be computed off-line. As shown in Table 1, the resulting visibility diagrams do not require excessive memory space to be stored, as they are represented just by a list of active constraints (borders) of each visibility region. The visibility information can therefore be easily stored and loaded along with the environment information.

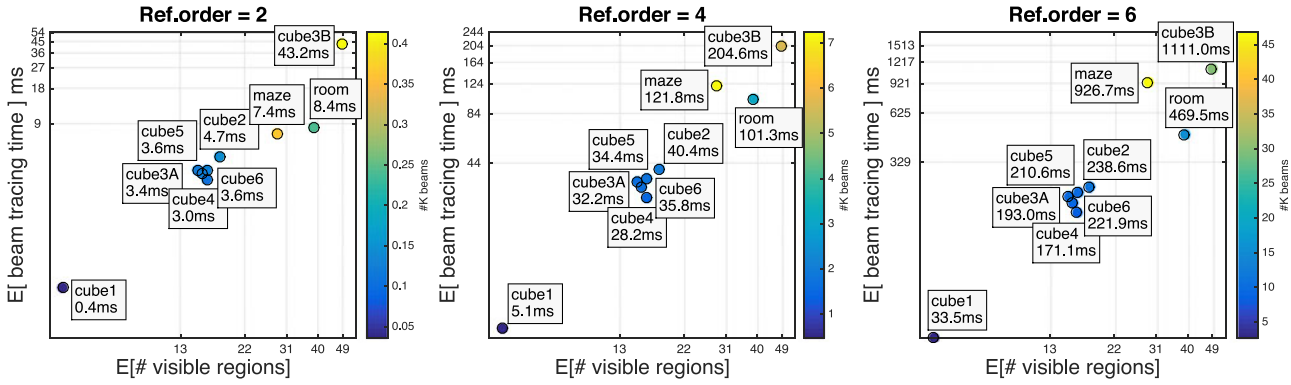
It is important to point out that unlike similar methods for computing the specular paths between the source and receiver, such as FastV [45] that efficiently computes a conservative P2R visibility and then uses it to speed-up P2P computation, our method computes the exact P2R visibility. As a consequence, the proposed method can be used not only to find the room impulse response between a source and receiver, but also to find the sound field contributions inside a spatially extended region. Such beam-like contributions can then be rendered using a loudspeaker array to reproduce the sound field of a virtual environment inside a listening area [30], [31].

4.3 Simulation Results

For each environment we run 1,000 Monte-Carlo simulations, in which source and receiver have been randomly placed within the environment. Intuitively, we expect the beam splitting and branching process to be more related to the visual complexity rather than to the geometrical one. The beam computation times, averaged across all source and receiver positions, are shown for the different types of environments for second, forth and sixth order of reflection in Figs. 9a, 9b, and 9c respectively. The x-axis represents the visual complexity of the environment and the colorbar indicates the average number of traced beams. The axis are in logarithmic scale for clarity of visualization.

TABLE 1
Time Needed to Precompute the Visibility Data Structure and Memory Required to Store it Together with the Environment Geometry Information

cube1	**cube2**	**cube3A**	**cube3B**	**cube4**
0.3 s 50 KB	58 s 290 KB	1 min 38 s 680 KB	1 h 18 min 1.6 MB	6 min 30 s 1.4 MB
cube5	**cube6**	**room**	**maze**	
11 min 2.3 MB	23 min 4.1 MB	35 s 55 KB	49 min 3.4 MB	



(a) (b) (c)

Fig. 9. Comparison of beam tracing times for different reflection orders.

Apart from the actual times that depend upon machine and implementation, the comparison between environments with varying complexities allows us to understand how the algorithm behaves with respect to geometry changes. From Fig. 9 we observe that the computational burden is nearly completely independent from the geometrical complexity and follows more closely the visual complexity. In fact, cube2, cube3A, cube4, cube5 and cube6 achieve, statistically, the same performance even if they are characterized by different geometrical complexities. As expected, cube3B is the computationally most expensive environment. The increase of the computational burden with visual complexity is approximately linear. In fact, while having higher visual complexity means more beam/region intersection tests, there is no need to consider interactions (occlusions) between regions.

Only in the maze, does our beam tracer behave slightly worse than expected due to visual complexity. The walls of the maze, in fact, are made of numerous small rectangles. In the maze environment, most of the walls are either completely visible or invisible. Because of this, the visibility regions of the maze are generally larger than those of cube3B. Our definition of visual complexity is, as a matter of fact, only approximate, as it does not account for the size of visibility regions, just their number. As beams bounce and split through the environment, they become smaller and are less-likely to hit small regions.

Table 2 shows more details on beam tracing and path determination times. More specifically, Table 2 shows average and standard deviation of beam tracing times, over all the realizations. Analogously, the average and standard deviation of path determination times are given. As expected, the beam tracing phase is more expensive than the path determination phase. In fact, intersecting a bundle of rays with a visibility region is computationally more demanding than testing if the receiver is inside a beam. The former is essentially an intersection between two convex linear regions, while the latter verifies if a point is inside a convex linear region. Furthermore, as expected, the standard deviation exhibits a higher value for environments with higher geometric complexity for both beam tracing and path determination phases. As an example, in the cube1 environment the choice of source and receiver locations does not impact on the number of traced beams. This fact is not true anymore in the bigger environments where the visibility changes considerably from location to location.

Finally, Fig. 10 shows the average computation time per beam at different reflection orders. We observe that the computation time per beam is higher at lower orders, and it gradually converges to a value that represents the average computation time of the beams traced through visibility lookup. This is due to the fact that, unlike the wall-reflected beams, the first (direct) beams originated from the acoustic source do not have a reference reflector and, therefore, can not be computed using the visibility information. As a consequence, to build a beam we not only need to test its incidence on a reflector but we also need to perform occlusion culling by iterating across all the potential occluders. In order to limit the number of potential occluders, our

TABLE 2
Beam Tracing and Path Determination Times

	Ref.	BT mean [ms]	BT std [ms]	PD mean [ms]	PD std [ms]
cube1	2	0.4	0.0	0.01	0.00
	4	5.1	0.2	0.16	0.01
	6	33.5	1.4	1.15	0.07
cube2	2	4.7	3.3	0.04	0.02
	4	40.4	24.6	0.68	0.37
	6	238.6	140.7	4.65	2.36
cube3A	2	3.4	3.2	0.04	0.02
	4	32.2	28.4	0.56	0.41
	6	193.0	172.8	3.95	2.90
cube3B	2	43.2	38.1	0.11	0.06
	4	204.6	147.1	1.93	1.15
	6	1111.0	754.3	12.26	6.83
cube4	2	3.0	3.8	0.03	0.02
	4	28.2	25.4	0.52	0.34
	6	171.1	150.5	3.56	2.38
cube5	2	3.6	5.3	0.03	0.02
	4	34.4	33.9	0.61	0.48
	6	210.6	200.4	4.20	3.13
cube6	2	3.6	5.2	0.04	0.03
	4	35.8	38.8	0.62	0.57
	6	221.9	237.5	4.27	3.66
room	2	8.4	0.7	0.07	0.01
	4	101.3	9.6	1.07	0.11
	6	469.5	44.1	6.75	0.66
maze	2	7.4	4.2	0.08	0.04
	4	121.8	91.7	2.20	1.42
	6	926.7	787.6	18.28	13.24

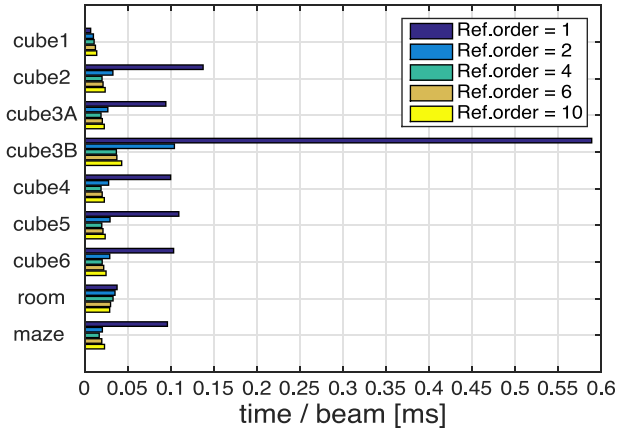


Fig. 10. Average computation time per beam at different reflection orders.

implementation of beam tracing from the acoustic source makes use of BSP. The wall-reflected beams, on the other hand, need to be tested only for incidence on precomputed visibility regions. The difference is more pronounced for cube3B environment, due to the more complex visibility conditions. Conversely, for the simplest environment, cube1, the visibility precomputation does not bring any relevant advantage since all the walls are visible without occlusions.

5 Conclusions

In this manuscript we proposed a generalization of the visibility-based beam tracing method [36] to the case of 3D geometry. This new definition of ray space, presented us with new challenges for defining visibility regions and, more generally, the data structure that encodes the R2R visibility in 3D space. We developed a solution for building beam trees through a lookup of such a visibility data structure, and tested it for computational efficiency, proving that it gives us the same advantages that its 2D counterpart [36] was able to offer, thus enabling a new class of solutions for interactive acoustic simulation in 3D space.

References

[1] A. Krokstad, S. Strøm, and S. Sørsdal, "Calculating the acoustical room response by the use of a ray tracing technique," *J. Sound Vib.*, vol. 8, no. 1, pp. 118–125, 1968.

[2] L. Savioja and U. P. Svensson, "Overview of geometrical room acoustic modeling techniques," *J. Acoust. Soc. Am.*, vol. 138, no. 2, pp. 708–730, 2015.

[3] W. Ahnert and R. Feistel, "EARS Auralization Software," in *Proc. Audio Eng. Soc. Conv.*, Oct. 1992.

[4] G. Naylor, "ODEON - another hybrid room acoustical model," *Appl. Acoust.*, vol. 38, no. 24, pp. 131–143, 1993.

[5] A. Farina, "RAMSETE - a new Pyramid Tracer for medium and large scale acoustic problems," in *Proc. EURO-NOISE*, Mar. 1995, p. 6.

[6] B.-I. L. Dalenbäck, "Room acoustic prediction based on a unified treatment of diffuse and specular reflection," *J. Acoust. Soc. Am.*, vol. 100, no. 2, p. 899, 1996.

[7] L. Savioja, J. Huopaniemi, T. Lokki, and R. Väänänen, "Creating interactive virtual acoustic environments," *J. Audio Eng. Soc.*, vol. 47, no. 9, pp. 675–705, 1999.

[8] J. J. Embrechts, "Broad spectrum diffusion model for room acoustics ray-tracing algorithms," *Acoust. Soc. Am. J.*, vol. 107, pp. 2068–2081, Apr. 2000.

[9] A. Silzle, H. Strauss, and P. Novo, "IKA-SIM: A system to generate auditory virtual environments," in *Proc. Audio Eng. Soc. Conv.*, May 2004, p. 6016.

[10] R. Kajastila, S. Siltanen, P. Lunden, T. Lokki, and L. Savioja, "A distributed real-time virtual acoustic rendering system for dynamic geometries," presented at the AES 122th Int. Conv. Pap. nr. 7160, Vienna, Austria, 2007.

[11] T. Lentz, D. Schröder, M. Vorländer, and I. Assenmacher, "Virtual reality system with integrated sound field simulation and reproduction," *EURASIP J. Appl. Signal Process.*, vol. 2007, no. 1, pp. 187–187, Jan. 2007.

[12] M. Noisternig, B. Katz, L. Savioja, and S. Siltanen, "Framework for real-time auralization in architectural acoustics," *Acta Acustica United Acustica*, vol. 94, no. 6, pp. 1000–1015, 2008.

[13] D. Schröder, *Physically Based Real-Time Auralization of Interactive Virtual Environments*,, vol. 11. Berlin, Germany: Logos Verlag Berlin GmbH, 2011.

[14] M. Taylor, A. Chandak, M. Qi, C. Lauterbach, C. Schissler, and D. Manocha, "Guided multiview ray tracing for fast auralization," *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 11, pp. 1797–1810, Nov. 2012.

[15] P. Annibale, F. Antonacci, P. Bestagini, A. Brutti, A. Canclini, L. Cristoforetti, E. Habets, J. Filos, W. Kellermann, K. Kowalczyk, A. Lombard, E. Mabande, D. Marković, P. Naylor, M. Omologo, R. Rabenstein, A. Sarti, P. Svaizer, and M. Thomas, "The SCENIC project: Space-time audio processing for environment-aware acoustic sensing and rendering," in *Proc. Audio Eng. Soc. Conv.*, Oct. 2011, p. 10.

[16] T. Betlehem and T. Abhayapala, "Theory and design of sound field reproduction in reverberant rooms," *J. Acoust. Soc. Am.*, vol. 117, no. 4, pp. 2100–2111, 2005.

[17] A. Canclini, D. Marković, F. Antonacci, A. Sarti, and S. Tubaro, "A room-compensated virtual surround system exploiting early reflections in a reverberant room," in *Proc. 20th Eur. Signal Process. Conf.*, 2012, pp. 1029–1033.

[18] D. Marković, K. Kowalczyk, F. Antonacci, C. Hofmann, A. Sarti, and W. Kellermann, "Estimation of acoustic reflection coefficients through pseudospectrum matching," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 22, no. 1, pp. 125–137, Jan. 2014.

[19] M. Vorländer, *Auralization: Fundamentals of Acoustics, Modelling, Simulation, Algorithms and Acoustic Virtual Reality*. New York, NY, US: Springer, 2007.

[20] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *J. Acoust. Soc. Am.*, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[21] J. Borish, "Extension of the image model to arbitrary polyhedra," *J. Acoust. Soc. Am.*, vol. 75, no. 6, pp. 1827–1836, 1984.

[22] U. Kristiansen, A. Krokstad, and T. Follestad, "Extending the image method to higher-order reflections," *Appl. Acoust.*, vol. 38, nos. 2–4, pp. 195–206, 1993.

[23] N. Tsingos and J. Gascuel, "A general model for simulation of room acoustics based on hierarchical radiosity," in *Proc. Visual Proc. ACM Comput. Graph.*, 1997, p. 2.

[24] E. Nosal, M. Hodgson, and I. I. Ashdown, "Improved algorithms and methods for room sound-field prediction by acoustical radiosity in arbitrary polyhedral rooms," *J. Acoust. Soc. Am.*, vol. 116, no. 2, pp. 970–980, 2004.

[25] M. Vorländer, "Simulation of the transient and steady-state sound propagation in rooms using a new combined ray-tracing/image-source algorithm," *Acoust. Soc. Am. J.*, vol. 86, pp. 172–178, Jul. 1989.

[26] T. Lewers, "A combined beam tracing and radiant exchange computer model of room acoustics," *Appl. Acoust.*, vol. 38, nos. 2–4, pp. 161–178, 1993.

[27] U. Stephenson and U. Kristiansen, "Pyramidal beam tracing and time dependent radiosity," in *Proc. 15th Int. Congress Acoust.*, Jun. 1995, pp. 657–660.

[28] G. I. Koutsouris, J. Brunskog, C.-H. Jeong, and F. Jacobsen, "Combination of acoustical radiosity and the image source method," *Acoust. Soc. Am. J.*, vol. 133, p. 3963, 2013.

[29] A. Canclini, D. Markovic, L. Bianchi, F. Antonacci, A. Sarti, and S. Tubaro, "A robust geometric approach to room compensation for sound field rendering," *IEICE Trans. Fundam. Electron., Commun. Comput. Sci.*, vol. E97.A, no. 9, pp. 1884–1892, 2014.

[30] F. Antonacci, A. Sarti, and S. Tubaro, "Two-dimensional beam-tracing from visibility diagrams for real-time acoustic rendering," *EURASIP J. Adv. Signal Process.*, vol. 1, pp. 1–18, 2010.

[31] D. Marković, A. Canclini, F. Antonacci, A. Sarti, and S. Tubaro, "Visibility-based beam tracing for soundfield rendering," in *Proc. IEEE Int. Workshop Multimedia Signal Process.*, Oct. 2010, pp. 40–45.

[32] P. S. Heckbert and P. Hanrahan, "Beam tracing polygonal objects," in *Proc. 11th Annu. ACM Conf. Comput. Graph. Interactive Techn.*, vol. 18, 1984, pp. 119–127.
[33] N. Dadoun, D. Kirkpatrick, and J. Walsh, "The geometry of beam tracing," in *Proc. 1st Annu. Symp. Comput. Geom.*, Jun. 1985, pp. 55–61.
[34] M. Monks, B. M. Oh, and J. Dorsey, "Acoustic simulation and visualization using a new unified beam tracing and image source approach," in *Proc. 101th Audio Eng. Soc. Conv.*, Nov. 1996, p. 8.
[35] J. Bittner and P. Wonka, "Visibility in computer graphics," *J. Environ. Planning*, vol. 30, pp. 729–756, 2003.
[36] F. Antonacci, M. Foco, A. Sarti, and S. Tubaro, "Fast Tracing of Acoustic Beams and Paths Through Visibility Lookup," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 4, pp. 812–824, May 2008.
[37] H. Fuchs, Z. M. Kedem, and B. F. Naylor, "On visible surface generation by a priori tree structures," in *Proc. 7th Annu. Conf. Comput. Graph. Interactive Techn.*, 1980, pp. 124–133.
[38] T. Funkhouser, I. Carlbom, G. Elko, G. G. Pingali, M. Sondhi, and J. West, "A beam tracing approach to acoustic modeling for interactive virtual environments," in *Proc. 25th Annu. ACM Conf. Comput. Graph. Interactive Techn.*, Jul. 1998, pp. 21–32.
[39] T. Funkhouser, P. Min, and I. Carlbom, "Real-time acoustic modeling for distributed virtual environments," in *Proc. 26th Annu. ACM Conf. Comput. Graph. Interactive Techn.*, 1999, pp. 365–374.
[40] F. Durand, G. Drettakis, J. Thollot, and C. Puech, "Conservative visibility preprocessing using extended projections," in *Proc. 27th Annu. Conf. Comput. Graph. Interactive Techn.*, 2000.
[41] S. Laine, S. Siltanen, T. Lokki, and L. Savioja, "Accelerated beam tracing algorithm," *Appl. Acoust.*, vol. 70, no. 1, pp. 172–181, 2009.
[42] C. Lauterbach, A. Chandak, and D. Manocha, "Interactive sound rendering in complex and dynamic scenes using frustum tracing," *IEEE Trans. Vis. Comput. Graph.*, vol. 13, no. 6, pp. 1672–1679, Nov. 2007.
[43] A. Chandak, C. Lauterbach, M. Taylor, Z. Ren, and D. Manocha, "AD-frustum: Adaptive frustum tracing for interactive sound propagation," *IEEE Trans. Vis. Comput. Graph.*, vol. 14, no. 6, pp. 1707–1722, Nov./Dec. 2008.
[44] R. Overbeck, R. Ramamoorthi, and W. R. Mark, "A real-time beam tracer with application to exact soft shadows," in *Proc. Eurograph. Symp. Render. Techn.*, 2007, pp. 85–98.
[45] A. Chandak, L. Antani, M. T. Taylor, and D. Manocha, "FastV: From-point visibility culling on complex models," *Comput. Graph. Forum*, vol. 28, no. 4, pp. 1237–1246, 2009.
[46] N. Raghuvanshi, J. Snyder, R. Mehra, M. Lin, and N. Govindaraju, "Precomputed wave simulation for real-time sound propagation of dynamic sources in complex scenes," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 68–78, Jul. 2010.
[47] D. L. James, J. Barbič, and D. K. Pai, "Precomputed acoustic transfer: Output-sensitive, accurate sound generation for geometrically complex vibration sources," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 987–995, Jul. 2006.
[48] N. Tsingos, "Pre-computing geometry-based reverberation effects for games," in *Proc. 35th AES Conf. Audio Games*, 2009, p. 10.
[49] N. Tsingos, T. Funkhouser, A. Ngan, and I. Carlbom, "Modeling acoustics in virtual environments using the uniform theory of diffraction," in *Proc. 28th Annu. Conf. Comput. Graph. Interactive Techn.*, 2001, pp. 545–552.
[50] 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," *J. Acoust. Soc. Am.*, vol. 115, pp. 739–756, 2004.
[51] U. P. Svensson, R. I. Fred, and J. Vanderkooy, "An analytic secondary source model of edge diffraction impulse responses," *J. Acoust. Soc. Am.*, vol. 106, no. 5, pp. 2331–2344, Nov. 1999.
[52] P. T. Calamia and U. P. Svensson, "Fast time-domain edge-diffraction calculations for interactive acoustic simulations," *EURASIP J. Appl. Signal Process.*, vol. 2007, no. 1, pp. 186–186, Jan. 2007.
[53] U. Stephenson, "Introducing higher order diffraction into beam tracing based on the uncertainty relation," *Building Acoust.*, vol. 18, nos. 1-2, pp. 59–82, 2011.
[54] A. Pohl, "Simulation of diffraction based on the uncertainty relation an efficient simulation method combining higher order diffractions and reflections," PhD dissertation, HafenCity Universität, Hamburg, Germany, 2013.
[55] S. Teller and M. Hohmeyer, "Determining the lines through four lines," *J. Graph. Tools*, vol. 4, no. 3, pp. 11–22, Nov. 1999.
[56] G. D. E. Fiume, "A fast shadow algorithm for area light sources using backprojection," in *Proc. 21st Annu. ACM Conf. Comput. Graph. Interactive Techn.*, 1994, pp. 223–230.
[57] S. J. Teller, "Computing the antipenumbra of an area light source," *Proc. 19th Annu. AMC Conf. Comput. Graph. Interactive Techn.*, vol. 26, no. 2, pp. 139–148, Jul. 1992.
[58] V. Koltun, Y. Chrysanthou, and D. Cohen-Or, "Hardware-accelerated from-region visibility using a dual ray space," in *Proc. 12th Eurograph. Workshop Rendering Techn.*, 2001, pp. 205–216.
[59] D. Sommerville, *Analytical Geometry of Three Dimensions*. London, U.K.: Univ. Press, 1934.
[60] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[61] J. Selig, *Geometrical Methods in Robotics*, ser. Monographs in computer science. New York, NY, US: Springer, 1996.
[62] S. Teller and M. Hohmeyer, "Stabbing oriented convex polygons in randomized $o(n^2)$ time," in *Proc. Jerusalem Combinatorics*, 1994, pp. 311–318.
[63] J. Stolfi, *Oriented Projective Geometry: A Framework for Geometric Computations*. San Francisco, CA, US: Academic, 1991.
[64] D. Cohen-Or, Y. Chrysanthou, C. Silva, and F. Durand, "A survey of visibility for walkthrough applications," *IEEE Trans. Vis. Comput. Graph.*, vol. 9, no. 3, pp. 412–431, Jul.–Sep. 2003.
[65] S. Nirenstein, E. Blake, and J. Gain, "Exact from-region visibility culling," in *Proc. 13th Eurograph. Workshop Rendering*, ser. EGRW '02, Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2002, pp. 191–202.
[66] D. Haumont, O. Makinen, and S. Nirenstein, "A low dimensional framework for exact polygon-to-polygon occlusion queries," in *Proc. Rendering Techn.*, 2005, pp. 211–222.
[67] F. Mora, L. Aveneau, and M. Mériaux, "Coherent and exact polygon-to-polygon visibility," in *Proc. Winter School Comput. Graph.*, 2005, pp. 87–94.
[68] S. Charneau, L. Aveneau, and L. Fuchs, "Exact, robust and efficient full visibility computation in Plucker space," *Vis. Comput.*, vol. 23, no. 9, pp. 773–782, 2007.
[69] F. Durand, G. Drettakis, and C. Puech, "The visibility skeleton: A powerful and efficient multi-purpose global visibility tool," in *Proc. 24th Annu. ACM Conf. Comput. Graph. Interactive Techn.*, 1997, pp. 89–100.

Dejan Marković was born in 1985. He received the MSc degree cum laude in 2009 and the PhD degree cum laude in 2013 from the Politecnico di Milano, Milan, Italy. Since January 2013, he is a post doctoral research assistant at the Image and Sound Processing Group (ISPG), Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano. His research activity is mainly focused on analysis, geometric modeling, and microphone array processing of acoustic wavefields for multimedia applications. He is a member of the IEEE.

Fabio Antonacci was born in Bari, Italy, in 1979. He received the Laurea degree, in 2004, in telecommunication engineering and the PhD in information engineering in 2008, both at the Politecnico di Milano, Italy. He is currently an assistant professor at the Politecnico di Milano. His research focuses on space-time processing of audio signals, for both speaker and microphone arrays (source localization, acoustic scene analysis, rendering of spatial sound) and on modeling of acoustic propagation (visibility-based beam tracing). He is author of more than 50 articles in proceedings of international conferences and on peer-reviewed journals. He is a member of the IEEE.

Augusto Sarti (M'04-SM'XX) received the MS and the PhD degrees in electronic engineering, both from the University of Padua, Italy, in 1988 and 1993, respectively, with research on nonlinear communication systems. His graduate studies included a joint graduate program with the University of California, Berkeley. In 1993, he joined the Politecnico di Milano, Milan, Italy, where he is currently an associate professor. In 2013, he also joined the University of California, Davis, as an adjunct professor. His research interests are in the area of multimedia signal processing, with particular focus on sound analysis, processing and synthesis; space-time audio processing; geometrical acoustics; music information retrieval. He also worked on problems of multidimensional signal processing, image analysis and 3D vision; motion planning and nonlinear system theory. He coauthored well over 200 scientific publications on international journals and congresses as well as numerous patents in the multimedia signal processing area. He coordinates the activities of the Musical Acoustics Lab and of the Sound and Music Computing Lab of the Politecnico di Milano. He promoted and coordinated or contributed to numerous (20+) European projects. He is a member of the IEEE Technical Committee on Audio and Acoustics Signal Processing, and Senior Editor of IEEE Signal Processing Letters. He was co-chairman of the 2005 Edition of the IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS); Chairman of 2009 edition of the Digital Audio Effects conference, (DAFx); and in the organizing committees of numerous other conferences in the area of signal processing. He is a senior member of the IEEE.

Stefano Tubaro (M'01-SM'XX) was born in Novara, Italy, in 1957. He received his Electronic Engineering degree at the Politecnico di Milano, Milano, Italy, in 1982. He then joined the Dipartimento di Elettronica e Informazione of the Politecnico di Milano, first as a researcher of the National Research Council; then as an Associate Professor (1991) and finally as a Full Professor (2004). He initially worked on problems related to speech analysis; motion estimation/compensation for video analysis/coding; and vector quantization applied to hybrid video coding. In the past few years, his research interests have focused on image and video analysis for the geometric and radiometric modeling of 3D scenes; and advanced algorithms for video coding and sound processing. He has authored over 150 scientific publications on international journals and congresses. He co-authored two books on digital processing of video sequences. He also co-authored several patents on image processing techniques. He coordinates the research activities of the Image and Sound Processing Group (ISPG) at the Dipartimento di Elettronica e Informazione of the Politecnico di Milano; which is involved in several research programs funded by industrial partners, the Italian Government, and by the European Commission. He has been involved in the IEEE Technical Committee of Multimedia Signal Processing (2005-2009), and is currently involved in that of Image Video and Multidimensional Signal Processing. He is a senior member of the IEEE.