A FRAMEWORK FOR THE DEVELOPMENT OF ACCURATE ACOUSTIC CALCULATIONS FOR GAMES

PANAGIOTIS CHARALAMPOUS, PANOS ECONOMOU

P.E Mediterranean Acoustics Research and Development, Limassol, Cyprus
panagiotis@pemard.com, panos@pemard.com

Despite the rapid development in acoustics calculation software during the last couple of decades, such advances have not been achieved uniformly. Various demands in different disciplines have shifted the focus to a number of different aspects of the calculations. Methods in game development have focused on speed and optimized calculation times to achieve interactive sound rendering, whilst engineering methods have concentrated in achieving accuracy for reliable predictions. This paper presents a flexible, expandable and adjustable framework for the development of fast and accurate acoustics calculations both for game development and engineering purposes. It decomposes the process of acoustic calculations for 3D environments into distinct calculation steps and allows third party users to adjust calculation methodologies according to their needs.

INTRODUCTION

Audio for games has progressed remarkably over the last years. There has been significant research on sound propagation algorithms and interactive sound rendering [1]. Solutions for interactive sound rendering have been proposed and relative implementations are widely available [2]. In parallel, methods which provide accurate predictions for acoustic phenomena have also been developed. Advanced calculations methods for accurately predicting various acoustics phenomena like sound reflection, diffraction, transmission and atmospheric absorption exist and have already been applied [3] [4] [5] [6] [7] [8] [9] [10]. It seems that these parallel developments took place with emphasis on different areas. This can be explained by the fact that each group was motivated by different concerns and aimed at different targets. Audio for game development focused on algorithms' interactivity and speed whilst acoustics software applications on accuracy at the expense of processing time [1]. Thus, accuracy has been considered as a trade-off for the one group whilst time, as a trade-off for the other.

Progress in hardware technology like multi core CPUs and programmable GPUs have exploited the possibility of accurate and realistic sound rendering for games. Accuracy and realism, no longer need to be considered as a trade-off and what was perceived as two distinct scientific areas can now converge into one unified scientific field for fast and accurate sound propagation calculations and rendering.

1 FRAMEWORK OVERVIEW

A software framework is an abstraction in which a software offering generic functionality can be selectively changed by user code, resulting in a specific software. A software framework is a universal, reusable software platform used to develop applications, products and solutions. The key distinguishing features of a framework are a) inversion of control b) default behavior c) extensibility d) non-modifiable framework code [11].

At the same time, separation of concerns (SoC) is a design principle that comprises of the process of separating a computer program into distinct features that overlap in functionality as little as possible. A Concern can be considered as any piece of interest or focus in a program. SoC is a principle that is important in the design of complicated systems and it is attributed to Edsger Dijkstra. As Dijkstra said, "the characteristic for all intelligent thinking is that one is willing to study in depth an aspect of one's subject matter in isolation for the sake of its own consistency, all the time knowing that one is occupying oneself only with one of the aspects" [12].

Using the SoC principle and focusing on the benefits of frameworks, we present a novel framework that can be used as a guiding baseline for the development of fast and accurate sound propagation calculations, which can be adjustable for both games and engineering applications. We call this framework PEMARD framework, after the initials of our company P.E. Mediterranean Acoustics Research & Development. PEMARD framework can be used as a guideline for parallel research in both calculations aspects, that of accuracy and speed.

PEMARD framework is a software architectural model which outlines a pattern that can be used in sound propagation calculations and defines a process for the calculation of sound propagation in 3D environments. It sets distinct steps in the process of sound propagation calculations by separating concerns in that process and designates interfaces between these steps. Also, it provides the infrastructure for the communication between these steps. As a result, the framework becomes a collection of several loosely coupled cohesive components which interoperate, based on well-defined interfaces for sound propagation calculations in a 3D environment. A loosely coupled model is one in which each of its components has, or makes use of, little or no knowledge of other separate components. Loose coupling results in the extensibility and modularity of the model and makes it ideal for customization based on needs. At the same time, it provides the interfaces which ensure the interoperability between different components from different research areas.

The framework defines four separate calculation steps. These are model optimization, pre-processing, sound propagation path detection and sound propagation path calculation. These steps are independent and communicate only through clearly defined interfaces. As a result, different implementations of each part can be combined based on the needs of different calculation methodologies.

The framework is based on geometrical acoustics for sound path detection. Geometrical acoustics can be defined as the description of sound propagation in terms of sound rays or sound paths. Geometrical acoustics is widely used in current commercial software applications which deal with acoustics calculations [8] [9] [10]. It is also the dominant approach in real time approaches for audio rendering in games [1][2] [13] [14].

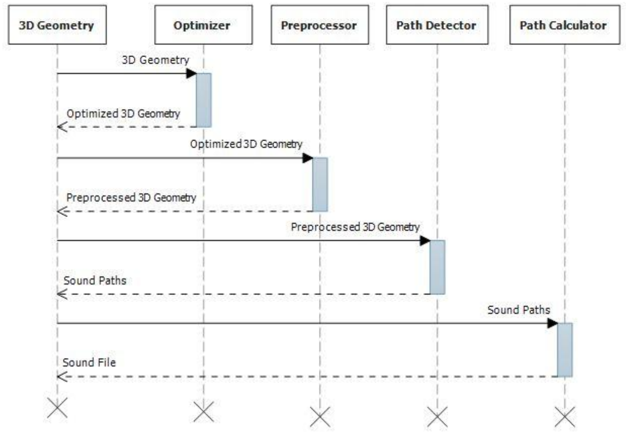


Figure 1: Sequence Diagram of PEMARD framework

Our framework can be demonstrated on a high level sequence diagram in Fig. 1. The demonstrated sequence diagram illustrates the relationships between the different components of the calculation process.

It is noted that the different components are loosely coupled. This means that the different components of the sequence diagram are only dependent on the underlying geometry object and the interfaces defined by it. Hence, they are independent from the processing taking place in other steps. As a result, the framework becomes flexible in interchanging the various components defined by it. To demonstrate this flexibility, an example could be that of a researcher who would like to experiment with a new path detection algorithm. The framework will provide him or her with the ability to replace the path detection component, without having to be concerned with the rest of the components.

The design of our software framework highlights three major pillars of research areas which are, model optimization, path detection and path calculation. These pillars are currently developed in parallel and can assist in realistic interactive sound rendering. Furthermore, we have included an additional step which allows extra custom processing on 3D geometries whenever this is required and possible. In the following paragraphs, we briefly describe each step of the calculation process.

2 OPTIMIZATION

Real time sound rendering algorithms are complex and resource consuming. Most reflection and edge diffraction detection methods have a complexity of $O(n^k)$, where n is the number of triangles in a geometry and k the order of reflection or diffraction required. Even other approaches like ray tracing, beam tracing and frustum tracing are directly dependent on the size of the 3D model being used [1][2] [13].

3D CAD or game models usually contain information relevant to graphics rendering which could be irrelevant to sound rendering. 3D environments contain a large amount of triangles in order to reproduce a realistic 3D visual environment. This amount of information is excessive for a realistic representation of a sound environment since sound wavelengths are considerably larger than those of light.

Differences in the nature of sound and light waves permit for simpler representations of 3D environments, allowing for shorter calculation times. Simplification of 3D meshes is a popular subject in current research, thus simplification methods for a 3D environment have been developed both for offline and real time processing. These methods can be applied in 3D sound rendering engines for enhanced performance [15] [16].

PEMARD’s framework allows the implementation and execution of a 3D model simplification and optimization step, which allows for adequate geometrical detail necessary for acoustical calculations. This results from the ability to plug-in this step into the process of calculation, either in real time or during the loading stage.

3 PREPROCESSING

The step of preprocessing is part of the workflow which is dependent on the methodology selected. For example, accurate analytical solutions require properties [4] which are not necessary for more simplified approaches [17]. Therefore, for the former type of calculations, extra information not present in raw 3D meshes, needs to be extracted from the model.

PEMARD framework provides the ability to execute or skip such a step depending on the methodology applied. Therefore, it enables the possibility of extracting metadata from a given geometry and using them in the preferred calculation process. It also allows for executing this step during loading or at any stage prior to any simulation, thus saving resources in real time calculation.

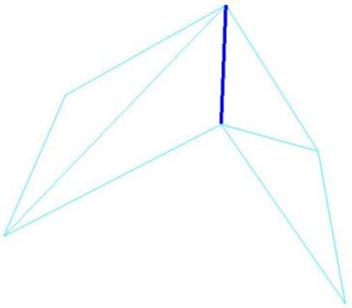


Figure 2: Edge on two triangles

An example of such metadata is the angles required for the calculation of the edge spherical diffraction coefficient based on the model of Hadden and Pierce [4]. A prerequisite for the determination of these angles is the knowledge of the triangles which form each edge (see Fig. 2). This information is extracted by a) processing a collection of 3D meshes b) determining the edges and c) finding which of the meshes share common edges. This is a time consuming process which is not suitable for a real time application.

4 SOUND PATH DETECTION

Sound path detection refers to the process of finding the sound paths from source to receiver. In Fig. 3 we can see an example of modeling sound propagation using Geometrical Acoustics to determine sound paths from a chiller to a microphone, around a noise barrier.

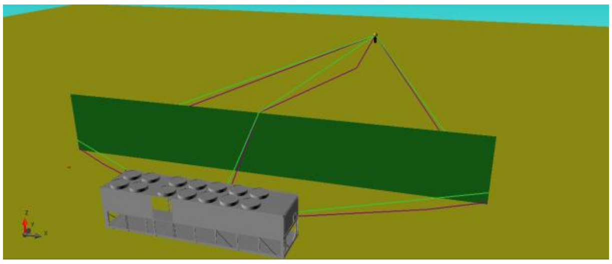


Figure 3: Sound paths between a chiller, a wall and a microphone

Sound path detection alongside with sound path calculation is the most intensive part of a sound rendering process. We can draw this conclusion by examining the complexity of the available detection algorithms [13]. Most of the known algorithms accurately detecting sound rays within a 3D environment, are exponential time algorithms. Even though these algorithms are used in non-interactive engineering purpose applications, their exponential time growth is a forbidding factor for real time calculations. As a result heuristic techniques have been developed for efficient sound propagation path detection, using ray tracing, beam tracing or frustum tracing techniques and visibility trees[1][2] [14] [18]. Although these methods can significantly speed up path detection times, they are compromising accuracy by missing important sound propagation paths.

The framework is designed to provide interchangeability of sound path detection methods based on needs. For example, let’s imagine a sound rendering engine based on the framework which has been developed based on a ray tracing method called X. At the same time a method named Y, which implements the framework interfaces, is published. Method Y performs significantly faster than X. The developer of the sound rendering engine does not need to rewrite the engine, but only needs to unplug from the process method X and plug into the process method Y.

5 SOUND PATH CALCULATION

Sound paths detected using one of the sound path detection methods are calculated and transformed to the corresponding frequency response or impulse response depending on the requirements of the calculation. Usually, sound rendering in 3D games requires the calculation and convolution of the impulse response with a prerecorded sound file.

Different methods exist for the calculation of the sound pressure level at a receiver. These methods vary in accuracy, speed and complexity. As a result, the selection of the method to be used is based on the requirements of the sound rendering process. For example, architectural acoustics and environmental noise mapping, require a high degree of accuracy employing advanced calculation methods. These methods take into account, angle dependent spherical wave reflection coefficients, finite impedance objects, finite size reflectors and edge corrections using Fresnel zones, high resolution frequency analysis, atmospheric turbulence and many more [10]. Gaming sound rendering engines cannot afford to calculate acoustic phenomena with this accuracy due to long processing times. As a result, they perform tradeoffs in calculations to be able to achieve interactive rates.

PEMARD framework uses the following general expression to calculate the relative pressure at each receiver, and thereafter, the sound pressure level.

$$p_{total} = \sum_{i=1}^{n} p_i \frac{e^{jkR_i}}{R_i} \prod_{j=1}^{m} C_j$$

Where

p_{total}, is the total sound pressure at a receiver, of all sound propagation paths from all sources,

p_i is the total sound pressure at a receiver, of all sound propagation paths from one source including barriers

n is the number of sound propagation paths from source to receiver

k is the wavenumber

R_i is the path length between a source and receiver

C_j is any coefficient that represents a sound phenomenon e.g. reflection, diffraction, atmospheric absorption etc.

m is the number of coefficients.

The basic operation the framework executes on a sound path is the attenuation due to distance. Thereafter, additional components can be added to the calculation in the form of coefficients. PEMARD framework is flexible in accepting an unlimited number of additional coefficients.

6 FRAMEWORK APPLICATION

We have used PEMARD framework to implement the simulation of sound propagation in 3D scenes.

This application is able to calculate direct sounds, reflected and diffracted sounds to any order as well as combinations of reflected-diffracted sounds. It detects sound paths using visibility tracing techniques alongside the image source method for reflections and an in-house developed algorithm for diffraction detection, based on the Broyden–Fletcher–Goldfarb–Shanno (BFGS) numerical optimization method. It calculates reflection coefficients for finite reflectors based on the spherical wave reflection coefficient and a correction for Fresnel zones [19] [20], diffractions based on Salomon's ray model for outdoor sound propagation [5], atmospheric turbulence and atmospheric absorption [21]. Below we provide a brief description of the sound calculation engine based on the framework steps described above.

6.1 Optimization

We optimize the geometry by removing unnecessary triangles from the 3D geometry. We use a probabilistic function which assumes that a collection of surfaces with the same vertices are adjacent to each other. The complexity of this method is $\mathrm{O(n)}$ where n is the number of surfaces. In the case of the accurate solution, the complexity would become.

$$O(\frac{(n+1)*n}{2}) \tag{3}$$

6.2 Preprocessing

We preprocess the geometry by detecting the distinct edges in the geometry and associating them with the meshes on which each edge finds itself. We need this information in order to be able to calculate the angles needed by the model described by Hadden and Pierce [4]. The complexity of this part is $O(xn)$ where x is the number of meshes in the geometry and n the number of edges in the geometry.

6.3 Path Detection

We detect reflected paths, diffracted paths and some cases of reflected-diffracted paths. We use an in-house path detection algorithm using conservative visibility tracing for reflection detection and in-house developed algorithm for diffraction detection based on the Broyden–Fletcher–Goldfarb–Shanno (BFGS) numerical optimization method. The complexity for the worst case scenario of detecting both reflection and diffraction remains $O(n^k)$, where k is the order of reflections or diffractions respectively. This applies only when all triangles are visible from all image sources. Such an extreme scenario is the case of sound propagation within a cube. In a real case scenario, the performance is better by orders of magnitude. Nevertheless, we have not been able to estimate an average performance of our tracing algorithm yet.

6.4 Path Calculation

We have implemented a path calculation model which utilizes sound ray modelling that solves Helmholtz's sound wave equation and thus accounts for sound diffraction to any order. Furthermore, it accounts for sound wave reflection from finite size surfaces of finite impedance using Fresnel Zones and spherical wave reflection coefficient concepts, respectively. The application uses flow resistivity for the calculation of the spherical wave reflection coefficient. Furthermore, it takes into account geometrical spreading, atmospheric absorption, and atmospheric turbulence. These embedded features allow the study of wave interference phenomena in resolutions down to single frequencies.

7 ENGINE APPLICATION

The engine described above is able to execute acoustical calculations for engineering purposes. An implementation of the engine was developed using C# and .Net and can yield almost interactive results on moderate hardware in relatively small geometries. Table 1 and Table 2 summarize performance results on a PC with a Core 2 Duo T6600 processor at 2.20 GHz. A geometry with 122 triangles and a geometry with 72 triangles were used for the benchmarking. Figure 4 and Figure 5 provide a visual representation of the two geometries.

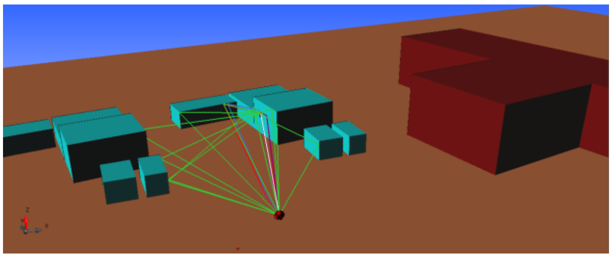


Figure 4: Geometry 1 with 122 triangles

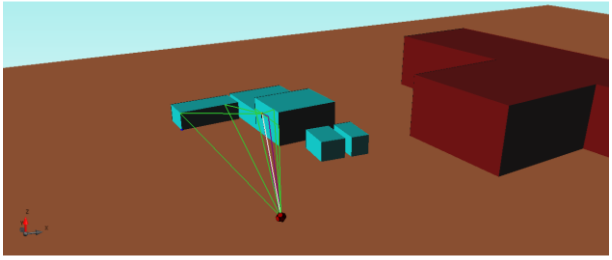


Figure 5: Geometry 2 with 72 triangles

Table 1 and Table 2 display the calculation time for different calculation settings for each geometry. The second column indicates the order of reflections detected. The third column indicates the order of diffractions detected. The fourth column indicates how many of the detected paths where taken into consideration for calculation. The paths are selected based on their indicative importance. For example, in setting 1, only the eight most important paths where calculated. The calculation time indicates the time required for the calculation of the frequency response in 1/3 octave bands. Each setting was calculated with a different combination of reflection and diffraction orders. Also a different number of paths were taken into consideration. From the table, we can extract the conclusion that the calculation of higher order of diffractions has a significant impact on the calculation time. This can be explained by the fact that the edges in a 3D environment are always more than the triangles; therefore more time is always required for diffraction than reflection detection of the same order.

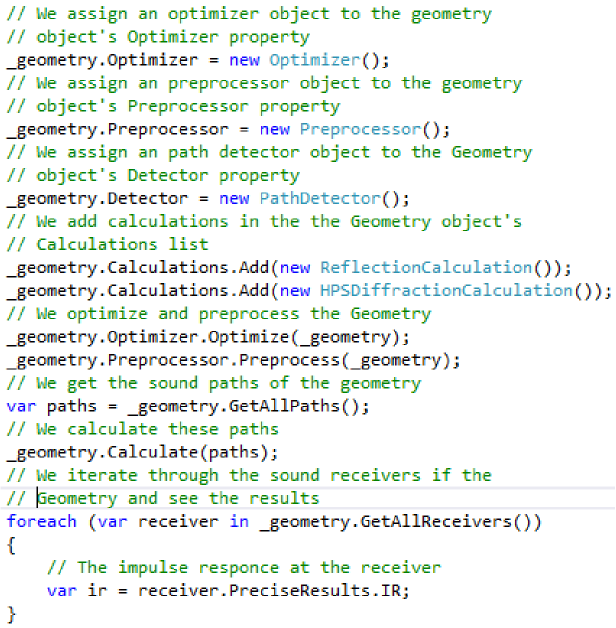
Table 1: Results for Geometry 1

Reflections Order	Diffractions Order	Paths Considered for Calculation	Time ms.
1	1	8	362
2	2	8	3452
4	2	16	3636

Table 2: Results for Geometry 2

Reflections Order	Diffractions Order	Paths Considered for Calculation	Time ms.
1	1	8	355
2	2	8	1687
4	2	16	1755

This specific implementation is suitable for engineering purposes but not appropriate for real time 3D sound rendering, since processing time is prohibitive for interactive purposes.



```
// We assign an optimizer object to the geometry
// object's Optimizer property
_geometry.Optimizer = new Optimizer();
// We assign an preprocessor object to the geometry
// object's Preprocessor property
_geometry.Preprocessor = new Preprocessor();
// We assign an path detector object to the Geometry
// object's Detector property
_geometry.Detector = new PathDetector();
// We add calculations in the the Geometry object's
// Calculations list
_geometry.Calculations.Add(new ReflectionCalculation());
_geometry.Calculations.Add(new HPSDiffractionCalculation());
// We optimize and preprocess the Geometry
_geometry.Optimizer.Optimize(_geometry);
_geometry.Preprocessor.Preprocess(_geometry);
// We get the sound paths of the geometry
var paths = _geometry.GetAllPaths();
// We calculate these paths
_geometry.Calculate(paths);
// We iterate through the sound receivers if the
// Geometry and see the results
foreach (var receiver in _geometry.GetAllReceivers())
{
    // The impulse responce at the receiver
    var ir = receiver.PreciseResults.IR;
}
```

Figure 6

In Figure 6, we can see a sample source code of a sample implementation of impulse response calculation for a specific geometry using the framework. We can see how we can assign to the geometry objects which are implemented based on predefined interfaces and calculate based on specific calculations. We can see the simplicity of setting up a sound propagation simulation engine by combining different independent components.

8 FRAMEWORK BENEFITS

PEMARD framework introduces the following benefits in the development of sound simulation engines. These are a) it outlines a pattern of a calculation process for acoustics simulations based on the principles of geometrical acoustics b) it provides the infrastructure for the cooperation of different implementations of various composing parts of the acoustic simulation process by defining distinct steps and clear interfaces between these steps and the framework c) By considering the design principle of separation of concerns and providing the ability to plug in existing implementations, the framework enables researchers to focus on their own part of research without being concerned with the other parts of the simulation process.

9 DISCUSSION

PEMARD framework does not aim to provide another solution in the problem of simulating sound for real time 3D environments. PEMARD framework serves the purpose of abstracting the calculation process for sound propagation in 3D spaces. It also allows researchers to focus on their specific scientific domain without having to be concerned how the rest of the infrastructure would develop. It follows the philosophy of generic software frameworks like .Net framework, MFC, NetBeans and Eclipse. Such an approach is novel in the niche world of acoustics software and acoustics simulations and can become the platform which can enhance and facilitate research and development of fast and accurate acoustic simulation engines.

10 CONCLUSION

Real time sound rendering in 3D spaces is a research subject covered by many individual approaches. Different researchers propose different methods bearing separate advantages and disadvantages. PEMARD framework is an approach which provides the infrastructure for systematic research in the field of real time sound rendering, the ability of combining different methodologies and the potential for further advancement in realistic game audio. PEMARD framework can serve as a platform for a joint development of fast and accurate sound propagation algorithms among scientists with different specializations. It can also provide the basis of commercial and research sound rendering applications.

REFERENCES

[1] A. Chandak, Efficient Geometric Sound Propagation Using Visibility Culling, Chapel Hill: University of North Carolina, 2011.

[2] C. Schissler and D. Manocha, "GSound: Interactive Sound Propagation for Games," in *AES 41st Conference: Audio for Games*, London, 2011.

[3] A. Pierce, "Diffraction of sound around corners and over wide barriers," *Journal of the Acoustical Society of America,* vol. 55, no. 5, pp. 941 -955, 1974.

[4] J. Hadden and A. Pierce, "Sound diffraction around screens and wedges for arbitrary point source locations," *Journal of the Acoustical Society of America,* vol. 69, no. 5, pp. 1266-1276, 1981.

[5] E. Salomons, "Sound Propagation in Complex Outdoor Situations with a Non-Refracting Atmosphere: Model Based on Analytical Solutions for Diffraction and Reflection," *Acta Acoustica,* vol. 83, pp. 436-454, 1997.

[6] P. Economou and P. Charalampous, "A comparison of ISO 9613 and advanced calculation methods using Olive Tree Lab-Terrain, an outdoor sound propagation software application: Predictions versus experimental results," in *Meeting on "Environmental Noise Propagation - Definitions, measuring and control aspects"* , London, 2012.

[7] A. Pierce, Acoustics - An introduction to its physical principles and applications., New York: Acoustical Society of America, 1994.

[8] "ODEON Room Acoustics Software," ODEON, [Online]. Available: http://www.odeon.dk/acoustics-simulation-software. [Accessed 2012].

[9] "EASE - Enhanced Acoustic Simulator for Engineers," AFMG, [Online]. Available: http://ease.afmg.eu/. [Accessed 2012].

[10] "OTL Terrain," PEMARD, [Online]. Available: http://www.otlterrain.com/. [Accessed 2012].

[11] D. Rielhe, Framework Design, Zurich: SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH, 2000.

[12] E. Dijkstra, "On the role of scientific thought," in *Selected writings on Computing: A Personal Perpective,* NY, Springer-Verlag, 1982, pp. 60-66.

[13] T. Funkhouser, N. Tsingos, I. Carlbom, G. Elko, S. Mohan, J. West, G. Pingali, P. Min and A. Ngan, "A beam tracing method for interactive architectural acoustics," *Acoustical Society of America,* vol. 115, no. 2, pp. 739-756, 2004.

[14] N. Tsingos and J. Wenyu, "Using programmable graphics hardware for acoustcs and audio rendering," *Journal of the Audio Engineering Society,* vol. 59, no. 9, pp. 628-646, 2011.

[15] A. Willmott, "Rapid simplification of multi-attribute meshes," in *Proceedings of the ACM SIGGRAPH Symposium on High Performance*

Graphics, New York, 2011.

[16] G. Humphreys, "Classification of Acceleration techniques," in *Image Synthesis*, University of Virginia, 2003.

[17] I. S. Organization, "Acoustics - Attenuation of sound during propagation outdoors - Part 2: General method of calculation," International Standards Organization, 1996.

[18] L. Antani, A. Chandak, M. Taylor and D. Manocha, "Efficient finite-edge diffraction using conservative from-region visibility," *Applied Acoustics*, vol. 73, pp. 218-233, 2011.

[19] C. S. Clay, D. Chu and S. Li, "Specular reflections on transient pressures from finite width plane facets," *Journal of the Acoustical Society of America,* vol. 94, no. 4, pp. 2279-2286, 1993.

[20] T. Embleton, J. Piercy and G. Daigle, "Effective flow resistivity of ground surfaces determined by acoustical measurements," *Journal of the Acoustical Society of America,* vol. 74, pp. 1239-1244, 1983.

[21] R. Nota, R. Barelds and D. Van Maercke, "Harmonoise WP 3 Engineering method for road traffic and railway noise adter validation and fine tuning," HARMONOISE, 2005.