Visualizing Acoustical Simulation Data in Immersive Virtual Environments

Sebastian Freitag, Dominik Rausch, Torsten Kuhlen

Virtual Reality Group, RWTH Aachen University
Seffenter Weg 23
52074 Aachen
Tel.: +49 (0) 241 / 80 - 24783
E-Mail: sebastian.freitag@rwth-aachen.de

Abstract: In this contribution, we present an immersive visualization of room acoustical simulation data. In contrast to the commonly employed external viewpoint, our approach places the user inside the visualized data. The main problem with this technique is the occlusion of some data points by others. We present different solutions for this problem that allow an interactive analysis of the simulation data.

Keywords: virtual reality, immersive visualization, acoustics, occlusion management

1 Introduction

For the design of a concert hall, a lecture room, or a theater, not only its visual appearance is important, but also—and sometimes mainly—its acoustical properties. These differ widely with the intended use of the room; e.g., while in concert halls, a full sound and a good spatial impression of music should be provided, the main priority in lecture rooms is the clear intelligibility of the lecturer throughout the room.

Nowadays, computer-based simulations have become more and more important as a helpful tool to describe room acoustics, as the computing power has increased to keep the time for a realistic simulation in an acceptable range; lately even allowing plausible real-time simulations. Visualizing these acoustical simulations has many advantages; for example, the designer can see the overall acoustic quality or certain attributes at several points in the room at once and distinguish them more clearly. Furthermore, the applications of room acoustical visualizations also include the analysis of simulation algorithms, teaching, or customer demonstrations.

In order to realize visualizations of room acoustics, there have been several approaches, most of them employing the classical two-dimensional screen as a display. In this paper, we examine the possibilities of a visualization of acoustical simulation data in an immersive virtual reality environment. In doing so, we concentrated on the employment of an inside-out perspective, allowing an analysis of the simulation data that is only possible with virtual reality techniques. In contrast to usually applied outside-in views, this perspective allows for an easier and more natural estimation of sizes and distances, since they are perceived

just as in the real world; and for which data points are more relevant than others, as the sense for reachable locations in the room is usually better. Furthermore, such a visualization could easily be added to any other simulation of a real architecture, preserving all sizes and proportions.

The main point of interest formulated by the domain experts was the visualization of single-valued attributes generated in a regular grid, corresponding to different acoustical qualities at that position (for details, see section 3.1) in a non-interpolated fashion. Consequently, for the examination of our approach, we used simulation data organized in a regular grid with a resolution of 1 meter along every axis (generated by the room acoustics simulation tool RAVEN [Sch10]), and concentrated on methods that allow for a detailed and undistorted inspection of these data points.

We will first present the current state of the art and related work in section 2. In section 3, we will introduce the visualization methods we used, and then describe the most important results of this contribution regarding the occlusion problem in section 4. We will finish with a short conclusion in section 5.

2 Related Work

There have been several approaches to visualize acoustical simulations in the past, either as a 2-D simulation and visualization, then usually ignoring the height of the room, or in three dimensions. Almost all of them use a 2-D desktop monitor as display.

In one of the first approaches for a working 3-D computer simulation and visualization of room acoustics, Stettner and Greenberg [SG89] introduced a system based on sound ray-casting, where at each of a number of pre-selected listener positions, a colored icon simultaneously displays up to twelve parameters measured at that point. Other visualization methods included ray diagrams showing the paths of the calculated sound rays, and the coloring of surfaces according to the received sound energy. Some properties necessary for an acoustical evaluation, like the comparison of different frequencies, were still missing.

Eduard Deines [Dei08, DMB+06] developed an algorithm to compute, and a prototype tool to visualize acoustical simulations based on a technique named *phonon tracing*. It works by simulating the path of virtual sound particles, called *phonons*, that approximate sound waves much like their visual counterparts, *photons*, approximate light waves. Apart from the rendering of individual phonons in movement—where the phonons were drawn as spheres, or discs—he also included the possibility to create and display isosurfaces to observe distinct wave fronts, among other methods.

The application of virtual reality techniques for the visualization of room acoustics has not been extensively explored yet. Lokki and Nenonen developed a prototype for a simple immersive visualization system based on a four-wall back-projection system [LN06]. They visualized ray paths and particles resulting from a ray-tracing simulation, using a simple user interface that allows a basic navigation through the scene, but did not include any further interaction methods.

3 Acoustic data simulation

In this section, we will introduce the general approach to visualize room acoustical simulation data. The prototype is designed for immersive virtual reality systems, especially *CAVE*-like systems. It is based on the *ViSTA VR toolkit* [AK08] and *VistaFlowLib* [SGvR+03] frameworks. ViSTA is a platform-independent open source toolkit for realizing virtual reality applications. The VistaFlowLib extends this framework by adding several different visualization methods.

3.1 Data

All tests were conducted using acoustical simulation data for the *Europa* hall of the Eurogress Aachen, generated using the simulation tool *RAVEN* [Sch10]. The acoustical simulation features different types of data sets, each of them sub-divided into 10 separate data sets for 10 different frequency bands representing the 10 octaves between 20Hz and 20kHz. In the visualization, the data sets energy (room impulse response in the first $500ms$ after the sound event), clarity $50ms$ (speech), clarity $80ms$ (music), definition, gain, center time, and reverberation time (for $-60dB$, $-30dB$ and $-20dB$) were examined. Most data sets under consideration were static, only the energy data set was time-varying over a number of discrete time steps. With a volume of about $20{,}000m^3$, and a resolution of one data point per cubic meter per frequency band per time step, and 100 time steps, this results in less than 100MBytes for the largest data set under examination. This amount of data presents no particular challenge for the visualization itself, as it can be stored completely in main or GPU memory. However, considering the number of data points to be visualized, information overflow can be an issue for the user. This is emphasized by the formulated requirement of having to be able to analyze all data points individually, and consequently requiring a discrete, non-interpolated display.

3.2 Basic interface design

The prototype uses a *flystick* as the only input device. The user can navigate through the virtual scene by moving physically or by using the cooliehat on the flystick, employing a point-and-fly metaphor. An additional option is to jump directly to the next room (if applicable).

All advanced user options are provided using menus and sliders. By pressing the corresponding buttons on the flystick, the user can activate two different menus that open in a half-circle floating in the air in front of the user (fig. **??**). Another button is used to select options and grab handles. A fourth button is used for quick interaction like turning on the *flashlight* (see section 4.2).

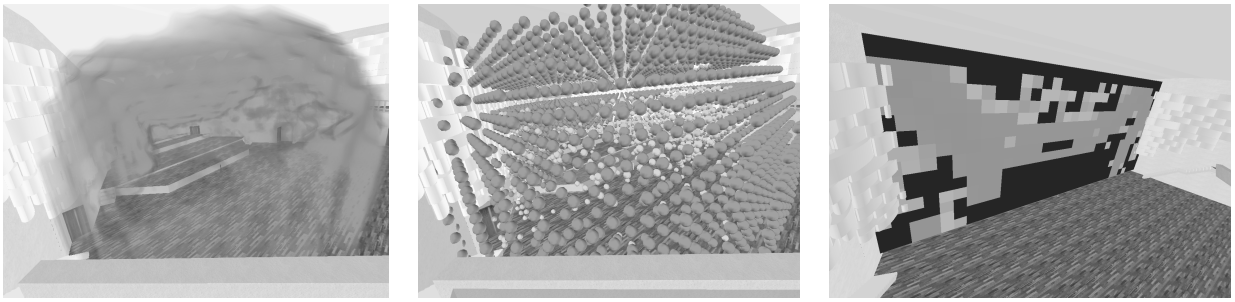

Figure 1: The different visualization methods: volume rendering, voxel rendering, cutplanes.

3.3 Visualization methods

For displaying volume data sets, three different visualization methods were examined here (see fig. 1):

Direct volume rendering

Here, the volume data set is rendered directly, using a slice-based approach. In the visualization, different colors and opacities represent different data values.

It turned out that direct volume rendering is no suitable method to visualize acoustical data sets. Firstly, they do not allow the analysis of discrete data points formulated as a requirement for the visualization. Even more importantly, the actual values of the data are hard to make out, as the different colors superimpose upon each other and mix. Using an inside-out perspective (standing in the data), all the user can see is a diffuse, multi-colored cloud.

Cutplanes

Cutplanes show a two-dimensional slice of the volume, and are often used when displaying a volume data set on a two-dimensional medium, like in print or on a screen. The usefulness of cut planes in virtual reality is limited, though, and cannot replace other visualization techniques. However, when viewing an animated scenery (like the energy distribution data set after a sound impulse), a non-moving cut plane can be used to visualize wave-fronts traveling through it. When placed on a wall, they illustrate the way a wave front hits that wall and is reflected. Other options include creating several cut planes and organizing them in a corner or even a cage around oneself, making it possible to visualize three dimensions of a wave front. Another possibility are cut planes that can be moved dynamically to convey a sense of the three-dimensional data grid (similar to sliding through the images in a computer tomography to get an idea of the internal body structures). The possibility to constrain the cutplanes to be orthogonal to one of the three main axes showed to facilitate the interaction and their interpretation.

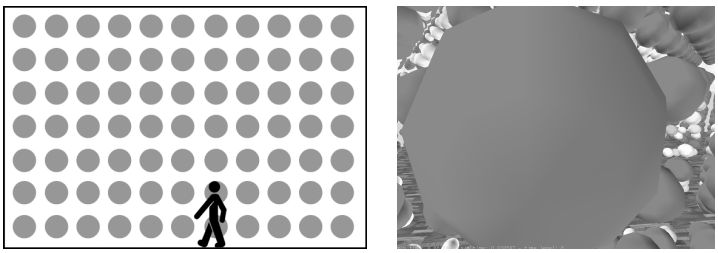

Figure 2: Close data points blocking the view nearly totally; schematic and ego-centric view.

Voxel rendering

The *voxel rendering* method renders the data values in each discrete room voxel as simple glyphs, i.e., each value is represented by a geometric object (like a cube or sphere) located in the center of the voxel, with the size and color dependent on the displayed value.

This method satisfies the requirements given, as the data values are displayed discretely and non-interpolated. It is easy to compare two individual points by looking at their color and size, the latter being especially helpful when looking down a row of points. However, the occlusion of some data points by others can be a problem, especially when employing an inside-out perspective, looking from within the data cloud. Several solutions to the occlusion problem were developed and are presented in section 4.

4 Occlusion handling

The goal of this work is to give the user a clear and intuitive view on the visualized simulation, while displaying as much of the data as possible at any point in time. This presents two challenges: one is a danger of information overflow for the user. Observing and interpreting a whole room of data points at once from within is a difficult task that needs support from the visualization system. The other is the problem of occlusion: some data points always block the view on others, disturbing an intuitive analysis. Both of these problems can generally be reduced to the question of when to display a data point and when not to; this issue will be addressed in this section.

In contrast to classical visualization approaches for 3-D data sets, virtually standing in the simulation intensifies the occlusion problem. Especially close data points can be problematic, as they usually block a large angle of view. The problem of standing within the scenery is that the simulated data points—visualized using the *voxel rendering* method, see section 3—block the user's sight on the rest of the scene, especially apparent with the ones close to the viewpoint (fig. 2). This problem is further intensified by the fact that the acoustical simulation data used here only has a low resolution—one data point per cubic meter of the room—since most acoustical characteristics differ only slightly in points very close to each other. This results in rather large shapes in each voxel, which means less space to look past.

Of course, the space between the data points could simply be enlarged by reducing the size

of the data point representations. However, there are several disadvantages to this approach. First of all, smaller shapes are harder to make out, and even in moderate distances, their size and color cannot be recognized with adequate accuracy. This effect is augmented by the limited resolution of the projectors used in the CAVE system, and especially apparent if the observer is standing close to a projection surface, allowing even less pixels for the remote data point. Another reason is that the maximum size of the shapes—representing the highest value—and therefore the value corresponding to the size of a certain point can be estimated much more easily if there is an easily identifiable frame of reference. If the maximum size corresponds to the voxel resolution (or slightly smaller), this frame of reference is simply the distance between two points. Moreover, even with a tolerably reduced size for the data point representations, there still would not be enough space for a good overview, as the gaps in between would still be considerably smaller than 1 meter. Lastly, the number of points in each data set is plainly too high for a comfortable examination if all of them are displayed.

Therefore, an approach to solve the occlusion problem is to hide a certain subset of the data points. Several rules to do this (that can also be employed in combination), called *volume selectors*, were developed, tested, and integrated into our prototype application. They will be introduced in this section, along with a discussion of their usefulness. For demonstration, we will only use graphics rather than screenshots to illustrate their function, as screenshots do not convey the effect obtained in an immersive visualization.

4.1 User sphere

Having identified data points very close to the head as the first problem (as demonstrated in fig. 2), a first approach is to hide everything within a certain distance from the head. This basic concept already enhances the visualization notably, since one can look around freely and still examine close data points (for small radii); the limits of the sphere can be easily conceived by moving a little, so confusing hidden data points with non-existent ones is not an issue.

4.2 Flashlight

Even with the user sphere, only very close data points can be seen comfortably; everything in a greater distance is still blocked by closer voxels. Furthermore, it is hard to see the context of the data or to find interesting spots in a distance. While navigating, the target and consequently an appropriate path to it cannot be seen, either.

A simple way to look at farther points would be to interactively increase the radius of the user sphere; however, that method is very inflexible and laborious, since its radius would have to be adjusted each time a data point at a different distance becomes the object of interest. Moreover, it would be impossible to see data points with different distances at the same time, and the problem of orientation while navigating would not be solved—either the way to the target would not be visible, or the target would be hidden while advancing to it.

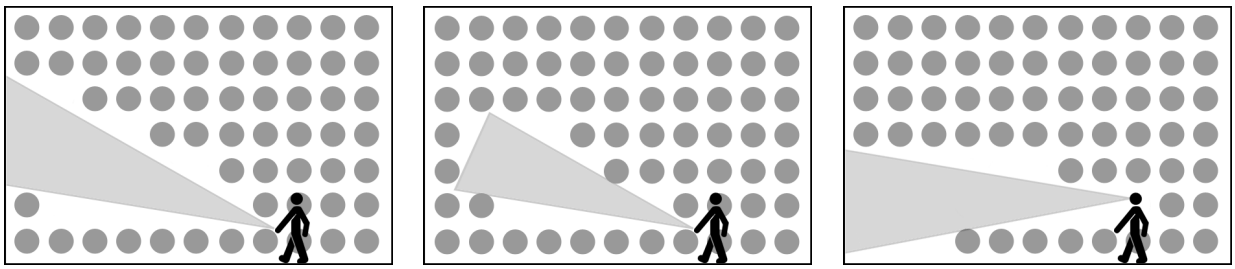

Figure 3: The hand-controlled flashlight; unlimited range, limited range, head-mounted.

Therefore, the *flashlight* metaphor has been introduced (see fig. 3). The user controls a cone with the control device—the end being at the hand position—affecting all voxels within. There are different ways to put this method into effect.

One way is to hide all voxels in an infinitely long cone-shaped beam with a certain opening angle. As the hand with the flashlight can be moved freely, the limits of the beam are easily recognizable, so confusing a hidden with a non-existing data point is not an issue. If a data point in some distance is to be observed, the flashlight can be held at an angle, hiding the points before and in the way of the interesting one. Since the flashlight can be controlled freely with one hand, it is a quite flexible instrument.

Another variant limits the range of the flashlight, hiding only points in a distance of no more than, e.g., 20 meters (fig. 3, middle). This can be helpful to quickly examine points at the other side of the room, and still allows a comfortable navigation, masking out data points on the way. However, it has shown that the range of the flashlight must not be fixed, and has to be adjustable at any time. We therefore included a function to interactively change the flashlight characteristics with an immediate preview of the result.

Further variations, including the inversion of the flashlight effect, or mounting it to the head (fig. 3, right) have proved to be less helpful. An inverted flashlight impedes the acquirement of a good overview of the data, as the context of the displayed data points cannot be seen. A head-mounted flashlight hides the data points in view direction, that is usually the direction of greatest interest to the user.

In practice, it has shown to be useful to let the actual flashlight beam start slightly behind the actual input device (about 1 meter). The reason for this is that the flashlight's cone is very small at its end, so in the vicinity of the user's hand it often shines just past a data point that occludes the view. This effect is amplified if the user holds the flashlight some way in front of him. Another possibility to correct this with a very similar impact would be to use an obtuse cone for the beam.

4.3 Explosion

While the flashlight is a good way to quickly find one's way through the data, or take a deeper look in a certain direction, it is not really qualified for getting a quick overview of the complete data set. Partly, this is because of the great number of data points displayed

at the same time. Therefore, the *explosion* metaphor employs a different approach, by automatically showing the data points in a short sequence. While the sequence is running, the user can then concentrate on interesting data points without any further input.

The user can trigger the explosion with a button on the input device. Then, a sphere starts to expand around his position that hides all data points within; comparable to an expanding user sphere. After having enclosed the whole room, the sphere starts to shrink again at a lower speed. Using a slower speed in the shrinking phase has shown to be useful, as one can often find a possibly interesting place in the expanding phase and re-examine it a little more carefully in the shrinking phase.

This volume selector allows to look at almost all data points, making them visible sequentially over a short period of time, where each point is visible to the user at least twice during the sequence. This not only allows to find interesting spots in the expanding phase and verify them in the shrinking phase, but also to estimate the distance to the interesting data points, as the expansion runs linearly and all data points in the first visible row have an identical distance.

In the worst case, this leaves only a very short period for some data points to be visible, so a close examination is generally not possible. The explosion is also unsuitable for viewing data sets that change over time, since some points will not be visible at all. It is a good instrument, though, for a quick overview over the data, for swiftly orientating oneself in the room, or for finding interesting spots.

4.4 Flashlight cannon

The *flashlight cannon* represents the conceptual combination of the flashlight and the explosion. It allows the user to fire slow flashlight beams with the input device. Similar to the explosion, the beams start with a low range that extends until reaching the final range, wait a little and then return backwards at a lower speed, following a similar rationale as the explosion. The flashlight cannon enables the user to selectively shoot several independent holes into the data cloud around him, allowing for a custom selection of a subset of the data. As the flashlight beam can have a limited range, it is also possible to compare distant values with closer ones by shooting one or multiple beams and comparing the far points with the ones unaffected by these beams.

Similar to the explosion volume selector, the flashlight cannon is less suitable for a detailed examination of certain data points, but rather to get an overview or quickly compare several hidden points; especially ones that are not immediately next to each other and thus harder to expose specifically using the other volume selectors.

As the primary use of this method is to examine the points at the end of the flashlight beam, the expansion speed can be chosen rather high, while the waiting time at full expansion should not be too short. In the tests, values around $4m/s$ for the expansion, $2m/s$ in the shrinking phase, and above $5s$ for the waiting time have shown to be useful.

4.5 Thinning

While the explosion can be useful to get a quick overview, its disadvantage is that it takes some time during which the observer is basically restricted to just watching. Therefore, another approach is not to hide all points over time, but to hide a significant amount of them at the same time without concentrating on a certain area.

The resulting method, called *thinning*, reduces the resolution of the displayed data points. This has two main advantages: for one thing, the free space between two displayed voxels becomes larger, resulting in a significantly clearer view on distant data points. A similar effect can also be achieved by reducing the size of the displayed data points; however, that way the data points themselves cannot be perceived as clearly. Moreover, the maximum space between voxels would also be limited to the basic resolution. Another advantage is that the number of visible data points is significantly decreased, thereby reducing the amount of information the user has to conceive at a time.

To avoid confusion, it is important to reduce the resolution uniformly, i.e., the same way on all axes, and regularly, leaving a constant distance between two displayed data points. For example, one could display only every second or third data point in each row. Higher values are generally not necessary, as showing every second data point already reduces the number of displayed points to one eighth (showing every third point means showing only one in 27 points).

This method turned out to be very useful for a comfortable overview of the data. Nearly a complete data set of the approx. 40m×15m×40m room that was tested could be seen at once, and also comfortably perceived in motion. It has to be kept in mind, though, that important areas are possibly hidden or less visible—in the worst case, when showing every second point in each row, from out of 27 interesting points in a $3 \times 3 \times 3$ cube only one is displayed. Hence, for a close examination, thinning should be turned off. Moreover, moving wave fronts, or generally time-dependent data, are harder to perceive with a lower resolution.

4.6 Moveable halfspaces

All previously introduced methods focus on gaining an overview of the data and an orientation within, and on finding data points for closer examination. Once the interesting spots are found, it can be useful to hide uninteresting parts of the data, or focus on a certain area. Defining such an area beforehand is very impractical, since usually that area is not known in advance, is different for different data sets and often not statically defined. Therefore, it has to be possible to define such areas interactively.

A tool for doing this are *moveable halfspace selectors*. They basically consist of a two-dimensional plane, where every data point on one side of it is hidden. If several halfspace selectors are in use, only points on the right side of *all* of them are hidden (*conjunction*), allowing to create an area of free space around oneself (see fig. 4, left).

An alternative possibility is to hide all points that are hidden by at least one of the halfspace selectors (*disjunction*, fig. 4, right). When several selectors are in use, this would

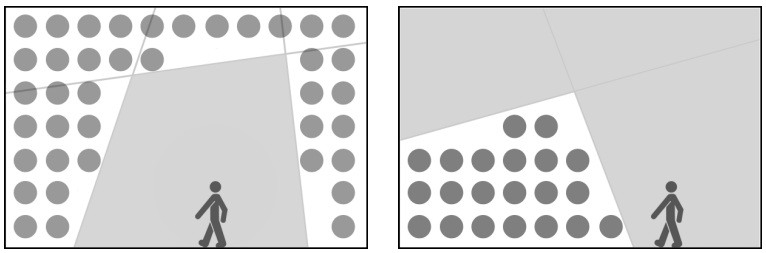

Figure 4: Moveable halfspaces, implemented as conjunction and disjunction.

result in forming an area with visible data points, while everything else becomes free space. In reality, though, this approach tends to hide too many data points. It also does not really make use of all of the advantages of virtual reality, as such an area usually only lies in one specific direction, reducing the area of interest to a much smaller angle. This approach rather corresponds to an outside-in, than an inside-out perspective.

In our visualization prototype, an arbitrary number of halfspace selectors can be interactively inserted, and moved or rotated by grabbing a handle mounted on the plane (for more details about the movement, see section 4.6.1). This way the free area can not only be created interactively, its borders can also be changed at any time, which is very useful to shift the area of interest.

4.6.1 Object and plane movement

To allow an interactive visualization, moveable halfspaces—as well as cutplanes—have to be interactive as well. A problem with these moveable planes is that the normally employed metaphor for moving objects in free space is to move the control device as far as one wants the controlled object to be moved—just like grabbing and moving a real object. However, this way the movement is limited to the area in arm's reach, plus the distance that can be walked in the VR environment; i.e., usually only a few meters. Thus, moving a plane through a room would require many single sequences of moving it a few meters.

However, just moving the target object by a multitude of the hand movement is not sufficient. Especially when the object is near, this results in considerable jittering. With cutplanes being very large objects, the jittering causes very unpleasant effects. Therefore, we developed a method for distance-dependent movement. There, the distance the object is moved is

$$d_{movement} = 4 \cdot d_{input} \cdot \sqrt{\max\{d_{handle}, 1\}} \tag{1}$$

where d_{input} is the distance the input device is moved from its position in the moment the handle was grabbed, and d_{handle} is the distance between the object's handle and the user. This method turned out to be a very comfortable way of moving cut planes and halfspace selectors, as it allows a movement through the whole room, while at the same time providing a good accuracy and low jittering when the plane is close to the user.

It showed that for planes, further restriction of the permitted degrees of freedom also helps. Therefore, the planes' orientation can be limited to the room's three primary axes, thereby considerably simplifying their movement.

4.7 Other volume selectors

In the course of this work, several other volume selectors have been developed, that cannot be discussed here in detail due to space limitations. These include

- hiding pre-defined areas of the room, dependent on where the user stands. This method has the disadvantage that meaningful partitions of the room have to be configured beforehand, which is usually before it is known which parts of the data set contain interesting values.
- opening a cross hallway on user command, pointing in view direction and to the sides in a right angle. The method can be a helpful tool if data along a specific axis of the room should be analyzed.
- hiding data points too far away from walls—assuming these are the less relevant positions, as usually there is no listener at these locations in the room. This method can be good for an overview, but it lacks flexibility, and often hides too many data points to detect certain trends.

5 Conclusion

In this contribution, an alternative approach for visualizing acoustical simulation data in immersive virtual environments was proposed. This method is different from previous approaches, as the data is not viewed from an external point of view, but from within the simulated scenery, using an inside-out perspective.

Several different visualization techniques were examined, among them Direct Volume Rendering, Voxel Rendering and interactive cutplanes. The use of an immersive environment allows for a better sense of the dimensions of the room and parts thereof, and proved to be a suitable method for visualizing acoustical simulation data.

However, one specific problem of this approach is the occlusion problem—when standing inside the data, far points are hidden by closer ones. To overcome this problem, we presented volume selectors that hide parts of the data in order to reduce the occlusion, get a better overview, and focus on specific points of interest. Several different volume selectors have been proposed and their usefulness discussed.

Our solutions have been integrated into an interactive visualization prototype designed for CAVE-like environments that allows an interactive selection of data sets and visualization methods, and to add or modify volume selectors. This prototype enabled an immersive and intuitive examination of the data, and showed the applicability of the inside-out visualization approach. While tailored for acoustical simulation data, our approach can also be extended to other types of low-resolution simulation data organized in a 3-D grid.

Still, some points remain for future research. One is the examination of multi-room sceneries and data sets to visualize the inter-dependencies between the acoustics of adjacent rooms, requiring different additional visualization and interaction techniques. Another possibility is the combination of auralization and visualization, especially as the former is already available for immersive environments [LSVA07]. Additionally, while we focused on single-valued acoustical attributes here, further aspects of research could also include the visualization of multiple values using more complex glyphs. As these show similar properties to the voxel rendering approach, they will also benefit from the techniques we developed.

References

[AK08] Ingo Assenmacher and Torsten Kuhlen. The ViSTA Virtual Reality Toolkit. In *Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, pages 23–26. Shaker Verlag, 2008.

[Dei08] Eduard Deines. *Acoustic Simulation and Visualization Algorithms*. Dissertation, TU Kaiserslautern, April 2008.

[DMB+06] E. Deines, F. Michel, M. Bertram, H. Hagen, and G. M. Nielson. Visualizing the Phonon Map. In Beatriz Sousa Santos, Thomas Ertl, and Ken Joy, editors, *EUROVIS - Eurographics /IEEE VGTC Symposium on Visualization*, pages 291–298, Lisbon, Portugal, 2006. Eurographics Association.

[LN06] Tapio Lokki and Ville Nenonen. Immersive Visualization of Room Acoustics. In *Joint Baltic-Nordic Acoustics Meeting*, November 2006.

[LSVA07] Tobias Lentz, Dirk Schröder, Michael Vorländer, and Ingo Assenmacher. Virtual Reality System with Integrated Sound Field Simulation and Reproduction. In *EURASIP Journal on Advances in Signal Processing*, 2007.

[Sch10] Dirk Schröder. *Physical based real-time auralization of interactive virtual environments*. Dissertation, RWTH Aachen University, 2010.

[SG89] Adam Stettner and Donald P. Greenberg. Computer Graphics Visualization For Acoustic Simulation. In *Computer Graphics (SIGGRAPH '89 Proceedings)*, pages 195–206, July 1989. Published as Computer Graphics (SIGGRAPH '89 Proceedings), volume 23, number 3.

[SGvR+03] M. Schirski, A. Gerndt, T. van Reimersdahl, T. Kuhlen, P. Adomeit, O. Lang, S. Pischinger, and C. H. Bischof. ViSTA FlowLib: A Framework for Interactive Visualization and Exploration of Unsteady Flows in Virtual Environments. In *7th International Workshop on Immersive Projection Technology, 9th Eurographics Workshop on Virtual Enviroments*, pages 077–086, Zurich, Switzerland, 2003. Eurographics Association.