A Multidimensional State-Space Approach for the Numerical Simulation of Sound Propagation in Enclosures

Sascha Spors and Rudolf Rabenstein
Telecommunications Laboratory
University of Erlangen-Nuremberg
Cauerstrasse 7, 91058 Erlangen, Germany
E-mail: {spors,rabe}@LNT.de

ABSTRACT

This paper presents a multidimensional state-space approach for the numerical simulation of sound propagation in enclosures. The simulation algorithm is essentially based on the wave digital filter principle however we will give a more direct access to the numerical solution of the wave equation here. To simulate sound propagation in enclosures, a detailed treatment of the boundary conditions is necessary. We focus in this paper on the treatment of memoryless boundary conditions in the new simulation algorithm.

1 INTRODUCTION

The design of real and virtual acoustical environments requires exact control of the sound field in enclosures. However, practical implementations such as concert halls, home theaters, car interiors, or sound rendering in computer games often fail to meet the user's expectations. On the one hand, the complexity of this task is considerable, since a number of acoustic channels are used to control amplitude and phase of sound waves in space and time. On the other hand, there are no suitable computer design tools. Most of the existing methods for the prediction of acoustical behaviour are based on the simplifying assumption that sound propagation can be modelled similar to the propagation of light waves. An exact simulation of a dynamic sound field is unfortunately only possible by numerically solving of the acoustical wave equation. For free space propagation, this is not a major problem, given that sufficient computing power is available. However, to simulate sound propagation in enclosures a detailed treatment of the boundary conditions is indispensable. In this paper we present a multidimensional state-space approach for numerical simulation of sound propagation in enclosures.

We proceed as follows: First we will introduce the core simulation algorithm an its modifications for boundary conditions in Section 2. Section 3 explains some details on the implementation and finally Section 4 shows some results using our proposed algorithm for the sound propagation simulation of a horn loudspeaker.

2 SIMULATION ALGORITHM

This section introduces the new algorithm for simulation of sound propagation in enclosures. In this paper we will concentrate on that part of the algorithm that handles the boundaries of the simulation enclosure. Nevertheless we will first shortly review the core algorithm.

2.1 Core Algorithm

The propagation of sound waves in air is governed by the equation of motion and the equation of continuity for the acoustic pressure $p(\mathbf{x},t)$ and the acoustic fluid velocity vector $\mathbf{v}(\mathbf{x},t)$ [1],

$$\rho_0 \frac{\partial}{\partial t}\mathbf{v}(\mathbf{x},t) + \text{grad}\; p(\mathbf{x},t) = \mathbf{e}_\text{s}(\mathbf{x},t) \tag{1a}$$

$$\frac{1}{\rho_0 c^2}\,\frac{\partial}{\partial t} p(\mathbf{x},t) + \text{div}\; \mathbf{v}(\mathbf{x},t) = j_\text{s}(\mathbf{x},t) \tag{1b}$$

where t denotes time and $\mathbf{x}$ the vector of space coordinates x, y, z. ρ_0 is the static density of the air and c is the speed of the sound. $\mathbf{e}_\text{s}$ and j_s are appropriate source terms. These two physical principles form a set of two coupled partial differential equations (PDEs) describing the propagation of sound waves. For our purposes a symmetric form of these equations is advantageous. This is achieved by introduction of the normalization constant $r_0 = \sqrt{3}\,\rho_0 c$ and combining (1) into one matrix equation

$$\underbrace{\begin{bmatrix} \rho_0 \mathrm{D}_t & 0 & 0 & r_0 \mathrm{D}_x \\ 0 & \rho_0 \mathrm{D}_t & 0 & r_0 \mathrm{D}_y \\ 0 & 0 & \rho_0 \mathrm{D}_t & r_0 \mathrm{D}_z \\ r_0 \mathrm{D}_x & r_0 \mathrm{D}_y & r_0 \mathrm{D}_z & 3\rho_0 \mathrm{D}_t \end{bmatrix}}_{\mathbf{Z}} \underbrace{\begin{bmatrix} i_1 \\ i_2 \\ i_3 \\ i_4 \end{bmatrix}}_{\mathbf{i}(\mathbf{x},t)} = \underbrace{\begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}}_{\mathbf{e}(\mathbf{x},t)} \tag{2}$$

where the operators $\mathrm{D}_t, \mathrm{D}_x, \mathrm{D}_y, \mathrm{D}_z$ denote partial derivation with respect to time and to the components x, y, z of $\mathbf{x}$. The components of $\mathbf{v}$ are denoted by i_κ, $\kappa = 1 \ldots 3$ and $i_4 = p/r_0$. Similarly, the components of $\mathbf{e}_\text{s}$ are denoted by e_κ, $\kappa = 1 \ldots 3$ and $e_4 = r_0 j_\text{s}$. This vector PDE is the starting point for the derivation of our simulation algorithm. It is essentially based on the multidimensional wave digital principle [2]. However, a more direct access is utilized here, based on a four-dimensional discrete-time and discrete-space state space description.

The derivation of this discrete system according to the state-space approach starts from the normalized vector PDE (2). After a series of intermediate steps, the state-space representation of a discrete-time and discrete-space algorithm is obtained as follows

$$\mathbf{z} = \mathcal{D}\left[\mathcal{A}\mathbf{z} + \mathcal{B}\mathbf{e}\right], \tag{3a}$$

$$\mathbf{i} = \mathcal{C}\mathbf{z} + \mathcal{F}\mathbf{e}. \tag{3b}$$

In this state space representation $\mathbf{z}$ denotes the internal state, $\mathbf{e}$ and $\mathbf{i}$ the input and output variables (acoustic pressure and particle velocity), $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, $\mathcal{F}$ are fixed matrixes and

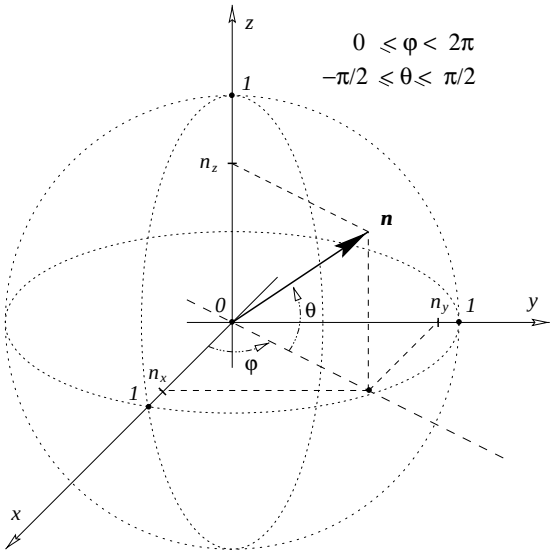


Fig. 1. Definition of the surface normal **n** for description of boundary surfaces

$\mathcal{D}$ is a operator matrix containing shifts in both directions of each spatial dimension as well as temporal delays. A detailed derivation of the simulation algorithm can be found in [3, 4].

The operator matrix $\mathcal{D}$ in the state equation (3a) contains shifts in both directions of each spatial dimension. This requires the knowledge of the previous states in all adjacent points. However, if a point lies at the boundary of the spatial domain, e.g. at the wall of an enclosure, one or more of the adjacent points are beyond the boundary, where the PDE is no more valid. The states of these points have to be determined from boundary conditions rather than from the PDEs (1). The next section introduces the modifications to the core algorithm which are necessary to handle the boundaries of an enclosure.

2.2 Boundary Point Classification

A boundary in the three-dimensional domain in our context is characterized by its location and its surface normal **n**

$$\mathbf{n} = \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} = \begin{bmatrix} \cos\varphi\cos\theta \\ \sin\varphi\cos\theta \\ \sin\theta \end{bmatrix}. \tag{4}$$

The surface normal is a vector of unit length which is orthogonal to the tangential plane of the surface. For our purposes the surface normal **n** is defined by the spherical coordinates φ and θ according to Figure 1. Unfortunately this definition cannot be used straightforward for our discrete space simulations because the boundary surface may not lie on the grid points. The boundary points have to be defined in conjunction with the spacial grid points. According to Figure 2 three types of points exist in our discrete simulation space: *interior points*, *boundary points* and *exterior points*. The *interior points* have interior or boundary points as direct neighbors. The spatial shifts in the operator matrix $\mathcal{D}$ can be carried out for all state components. These points are handled by the core algorithm described in the previous

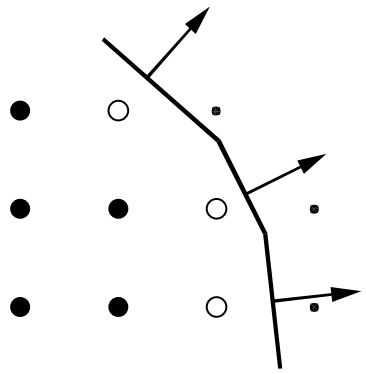


Fig. 2. Types of grid points: • interior points, ∘ boundary points, · exterior points

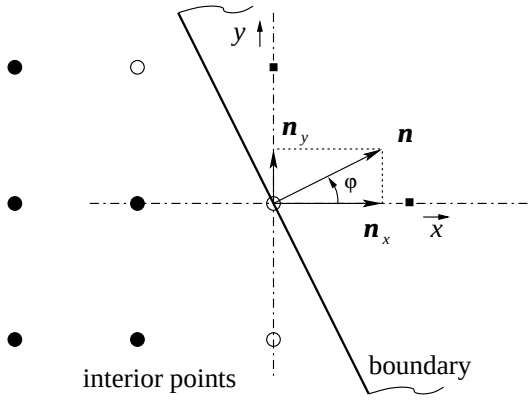


Fig. 3. Projection of the surface normal **n** in the spatial discrete simulation grid. For simplicity only 2 dimensions are shown here.

section. A point is a *boundary point* if it has at least one neighbor point outside the boundary (exterior point). Thus one or more of the shift operations in the operator matrix $\mathcal{D}$ cannot be carried out and the missing state components have to be calculated from the boundary conditions. *Exterior points* lie outside of the simulation space and therefore they do not have to be considered at all. The core algorithm has to be extended to handle the boundary points correctly. According to Figure 3 the existence of a neighbor point depends only on the direction of the μ-th component n_μ of the surface normal **n** with $\mu \in (x, y, z)$. The state components of the boundary points can be classified into two classes: *existent state components* and *missing state components*. Denoting the ν-th component of the state vector **z** with $\mathbf{z}_\nu$ there are three possibilities for $(\mu, \nu) \in \{(x, 1), (y, 2), (z, 3)\}$

1. $n_\mu > 0$: The upper neighboring point in μ direction lies outside. As a cause the component $z_{2\nu-1}$ cannot be calculated out of the state equation. This state component is labeled as *missing state component*.
2. $n_\mu = 0$: The two neighboring points lie on the boundary surface. Therefore they can deliver the components $z_{2\nu-1}$ and $z_{2\nu}$ of the state vector.
3. $n_\mu < 0$: The lower neighboring point in μ direction lies

outside. As a cause the component $z_{2\nu}$ cannot be calculated out of the state equation. This state component is labeled as *missing state component*.

Discarding the case when all state components can be calculated directly there are overall $3^3 - 1 = 26$ possibilities for three spatial dimensions.

2.3 Boundary Conditions

The relation between the acoustic pressure p and the acoustic fluid velocity component v_n perpendicular to the boundary can be expressed by the acoustic resistance R_w [5] in continuous space as follows

$$p = R_w v_n = R_w \mathbf{n}^T \mathbf{v} \tag{5}$$

for a memoryless boundary. We restrict the investigations in this paper to memoryless boundary conditions but the algorithm can also be extended to frequency dependent surface reflection. The acoustic resistance R_w can be defined in terms of the real valued surface reflection factor r as follows

$$R_w = \rho_0 c \, \frac{1+r}{1-r}. \tag{6}$$

Equation (5) can be rewritten in the variables used in the continuous vector PDE (2)

$$r_0 i_4 = R_w \mathbf{n}^T \begin{bmatrix} i_1 \\ i_2 \\ i_3 \end{bmatrix}. \tag{7}$$

The basic idea of the modified algorithm for boundary points is to split each boundary point into three virtual boundary points with the same spatial location but surface normals parallel to the x, y, z axis. This permits separate computation of the missing states. Figure 3 shows a example for the projection of the surface normal $\mathbf{n}$. Using Equation (4) we can rewrite Equation (7) into its spatial projected components

$$r_0 i_4 = \cos\varphi \cos\theta R_w i_1, \tag{8a}$$
$$r_0 i_4 = \sin\varphi \cos\theta R_w i_2, \tag{8b}$$
$$r_0 i_4 = \sin\theta R_w i_3. \tag{8c}$$

2.3.1 Existing State Components

The existing state components are combined into the state vector $\mathbf{z}_\mathrm{i}$ with the dimension $(e \times 1)$ with $e = 3, 4, 5$. The existing boundary states can be calculated from the state vector $\mathbf{z}$ with the help of a suitable transformation matrix $\mathbf{T}_\mathrm{i}$ containing only ones and zeros.

$$\mathbf{z}_\mathrm{i} = \mathbf{T}_\mathrm{i}^T \mathbf{z} \tag{9}$$

The transformation matrix selects the appropriate state components and is therefore dependent on the geometry of the enclosure. For an efficient implementation the transformation matrix $\mathbf{T}_\mathrm{i}$ can be calculated in advance for all 26 possibilities of boundary geometries.

2.3.2 Missing State Components

Similar as for the existing state components, the missing state components are combined into the state vector $\mathbf{z}_\mathrm{b}$ with the dimension $(n \times 1)$ with $n = 1, 2, 3$. The missing state components can also be calculated from the state vector $\mathbf{z}$ with the help of a suitable transformation matrix $\mathbf{T}_\mathrm{b}$ similar to $\mathbf{T}_\mathrm{i}$.

$$\mathbf{z}_\mathrm{b} = \mathbf{T}_\mathrm{b}^T \mathbf{z} \tag{10}$$

The transformation matrixes have the following property

$$\mathbf{T}_\mathrm{i}\mathbf{T}_\mathrm{i}^T + \mathbf{T}_\mathrm{b}\mathbf{T}_\mathrm{b}^T = \mathbf{I} \tag{11}$$

which can be used together with (9) and (10) to derive the state vector $\mathbf{z}$ from the vectors $\mathbf{z}_\mathrm{i}$ and $\mathbf{z}_\mathrm{b}$

$$\mathbf{z} = \left(\mathbf{T}_\mathrm{i}\mathbf{T}_\mathrm{i}^T + \mathbf{T}_\mathrm{b}\mathbf{T}_\mathrm{b}^T\right)\mathbf{z} = \mathbf{T}_\mathrm{i}\mathbf{z}_\mathrm{i} + \mathbf{T}_\mathrm{b}\mathbf{z}_\mathrm{b} \tag{12}$$

The missing states $\mathbf{z}_\mathrm{b}$ have to be calculated from the boundary conditions as follows

$$\mathbf{z}_\mathrm{b} = \mathcal{A}_\mathrm{b}\mathbf{z}_\mathrm{i} + \mathcal{B}_\mathrm{b}\mathbf{e} \tag{13}$$

The matrixes $\mathcal{A}_\mathrm{b}$ and $\mathcal{B}_\mathrm{b}$ can be calculated using equation (8).

2.4 Modified State-Space Description

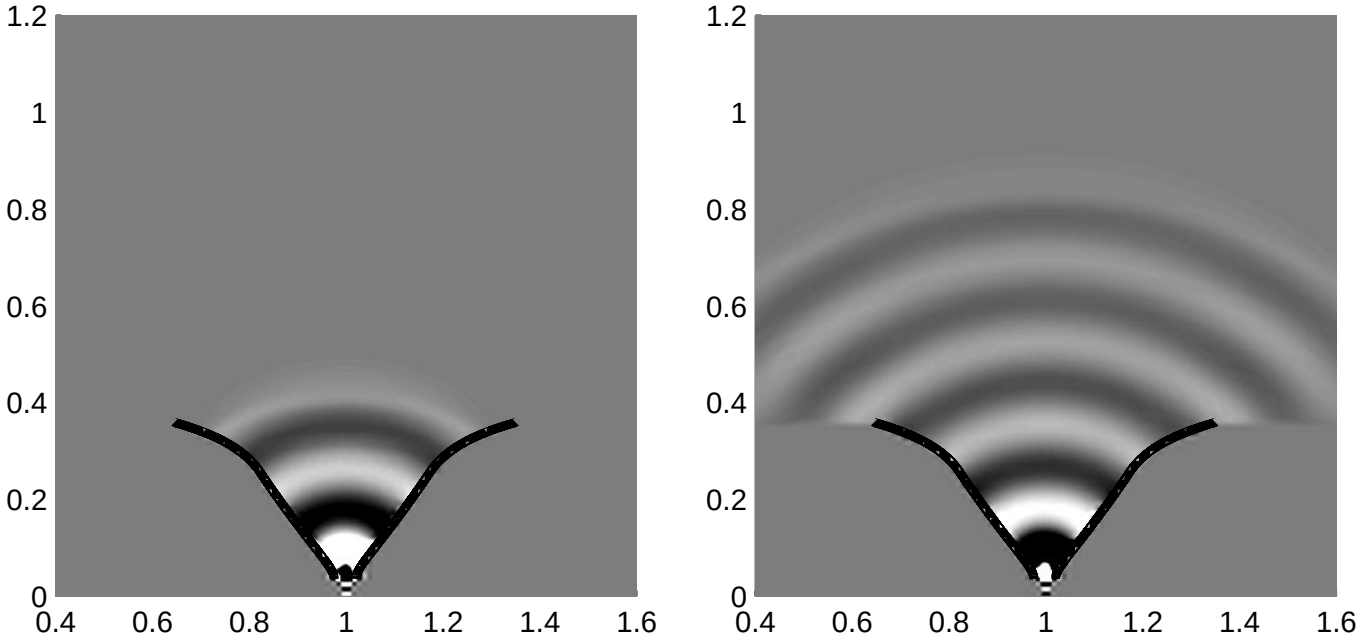
The algorithms described for existing and missing state components can be merged into one modified state space description of the whole algorithm. The existing state components follow from a state equation similar to (3a). The missing state components follow from the interior states and the boundary conditions. The existing and missing state components are therefore also called interior and boundary state components. The state space representation has to consider both types of states appropriately. Its general form is given by

$$\mathbf{z}_\mathrm{i} = \left(\mathbf{T}_\mathrm{i}^T \mathcal{D}\right)\left[\mathcal{A}\mathbf{z} + \mathcal{B}\mathbf{e}\right], \tag{14a}$$
$$\mathbf{z}_\mathrm{b} = \mathcal{A}_\mathrm{b}\mathbf{z}_\mathrm{i} + \mathcal{B}_\mathrm{b}\mathbf{e}, \tag{14b}$$
$$\mathbf{z} = \mathbf{T}_\mathrm{i}\mathbf{z}_\mathrm{i} + \mathbf{T}_\mathrm{b}\mathbf{z}_\mathrm{b}, \tag{14c}$$
$$\mathbf{i} = \mathcal{C}\mathbf{z} + \mathcal{F}\mathbf{e}. \tag{14d}$$

The matrices $\mathbf{T}_\mathrm{i}$ and $\mathbf{T}_\mathrm{b}$ depend on the geometry and describe whether a state is an interior state $\mathbf{z}_\mathrm{i}$ or a boundary state $\mathbf{z}_\mathrm{b}$. Equation (14a) is very similar to the state equation (3a), except that it delivers only the interior states. The boundary states are computed in (14b) from the interior states and the boundary conditions, which determine $\mathcal{A}_\mathrm{b}$ and $\mathcal{B}_\mathrm{b}$. Both interior and boundary states are merged into the complete state vector $\mathbf{z}$ in (14c). It is used to deliver the output quantities in (14d) and to update the interior states in (14a).

3 IMPLEMENTATION

The algorithm described above has been implemented in `C++` in an object oriented fashion. This implementation is based on a multidimensional systems library which was developed at our laboratory. In the current version, objects with rectangular shape and analytical objects of 2nd degree (e.g. ellipsoids) with given surface reflexion factor can be modeled. Available sources include point sources, loudspeaker arrays and horn loudspeakers. The respective sound pressure of the wavefield can be captured at any point within the spatial grid.



(a) $t = 1.3$ ms

(b) $t = 2.5$ ms

Fig. 4. Snapshots of the simulated wavefield for the modified OS waveguide. The horn was exited with an sinusoidal signal starting at $t = 0$ at the throat of the horn.

4 RESULTS

To show the performance of the algorithm described above, we computed the wave propagation for an complex geometry. The experimental setup consists of an horn loudspeaker placed at one wall of an room which has the size $1.8 \times 1.8 \times 1.8$ m (w × d × h). All surrounding walls of the room have a surface reflection factor of $r = 0$ and are therefore absorbing. This way free field conditions are modelled. The geometry of the horn loudspeaker is a modified oblate spheroidal (OS) waveguide. The surface of the horn loudspeaker has an surface reflection factor of $r = 0.99$. The simulation gives insights into the wave propagation inside and outside the horn loudspeaker and helps to optimize the design of horn loudspeakers. The simulation was performed in three-dimensions with the algorithm described in this paper, especially the boundary conditions where treated as outlined in the sections above. In order to visualize the wavefield we captured the acoustic sound pressure at one cutting plane through the horn loudspeaker. Figure 4 shows snapshots of the wavefield at $t = 1.3$ ms and $t = 2.5$ ms after for an sinusoidal excitation starting at $t = 0$ at the throat of the horn. The results show that numerical simulation of acoustic wave propagation through the proposed algorithm is able to reproduce the physical effects of transmission, reflection and diffraction. The simulation of the modified OS waveguide shows also the benefits of this geometry clearly. The results show that the diffraction effects caused at the mouth of an unaltered OS waveguide are nearly suppressed by the modified geometry. Using our method it is also possible to calculate the frequency response of the loudspeaker at each point within the room. This can simply be done by using a Dirac impulse for excitation of the horn and recording of the impulse response somewhere in the room.
Other examples and scenarios can be found in [6].

5 REFERENCES

[1] K.H. Kuttruff, "Sound in enclosures," in *Handbook of Acoustics*, M.J. Crocker, Ed. John Wiley and Sons, Inc., 1998.

[2] A. Fettweis, "Multidimensional wave-digital principles: From filtering to numerical integration," in *Proc. Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP 94)*. IEEE, April 1994, pp. VI–173 – VI–181.

[3] R. Rabenstein and A. Zayati, "Sound field simulation by computational acoustics. Part I: Simulation algorithm," *Int. Journal of Adaptive Control and Signal Processing*, vol. 14, pp. 663–680, 2000.

[4] S. Spors and R. Rabenstein, "Characterization of acoustical environments by numerical simulation," in *7th International Workshop on Acoustic Echo and Noise Control (IWAENC)*, Darmstadt, Germany, 2001, pp. 195–198.

[5] L. J. Ziomek, *Fundamentals of Acoustic Field Theory and Space-Time Signal Processing*, CRC Press, Boca Raton, 1995.

[6] `http://www.LNT.de/~spors/WPSIM`.