EUROGRAPHICS 2004 / M.-P. Cani and M. Slater
(Guest Editors)

Volume 23 (2004), Number 3

Synthesizing Sound from Turbulent Field using Sound Textures for Interactive Fluid Simulation

Yoshinori Dobashi[1] Tsuyoshi Yamamoto[1] Tomoyuki Nishita[2]

[1]Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan
[2]Graduate School of Frontier Science, The University of Tokyo, Tokyo, Japan
{doba, yamamoto}@nis-ei.eng.hokudai.ac.jp, nis@is.s.u-tokyo.ac.jp

Abstract

Sound is an indispensable element for the simulation of a realistic virtual environment. Therefore, there has been much recent research focused on the simulation of realistic sound effects. This paper proposes a method for creating sound for turbulent phenomena such as fire. In a turbulent field, the complex motion of vortices leads to the generation of sound. This type of sound is called a vortex sound. The proposed method simulates a vortex sound by computing vorticity distributions using computational fluid dynamics. Sound textures for the vortex sound are first created in a pre-process step. The sound is then created at interactive rates by using these sound textures. The usefulness of the proposed method is demonstrated by applying it to the simulation of the sound of fire and other turbulent phenomena.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism - Animation; I.6.8 [Simulation and Modeling]: Types of Simulation - Animation; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism; I.6.3 [Simulation and Modeling]: Applications; H.5.5 [Information Interfaces and Presentation]: Sound and Music Computing - Methodologies and techniques, Modeling.

1. Introduction

The simulation of sound effects for realistic virtual environments has become an important research field, even in the computer graphics community. In the past decade, a lot of methods have been developed for creating sound effects by simulating the source and propagation of sound [TH92][HGL*95][OCE01][Coo02][DBJR*02]. Several recent methods make it possible to simulate sound in real-time [FMC99][vdDKP01][TFNC01][OSG02][LSV*02][DYN03]. These methods can create realistic sounds that relate to objects in motion, and which depend on the geometric relationship between the receiver and the sound source.

Most of the previous research into the sound synthesis focused on the sounds generated by subtle oscillations of solid objects, such as the sound due to objects colliding. In 2003, however, Dobashi et al. proposed an interesting method for simulating the sound created by the motion of fluids [DYN03]. Examples of sounds generated by fluids include the sound of wind, fire, water and so on. Dobashi et al. succeeded in a realistic simulation of the sound generated by swinging swords. This method can only simulate the sounds that are generated when objects are present in a flow. However, even if no objects are in the fluids, sound is generated from a turbulent field due to the motion of fluids itself. For example, we can hear the sound from a fire or from a jet stream. This is caused by the motion of vortices in the fluid and is called a *vortex sound* [How03].

The aim of this paper is to simulate vortex sounds corresponding to the motion of fluids obtained by the numerical simulation at interactive rates. There are two possible approaches to simulate the sound. One of these approaches analyzes a collection of recordings to extract the significant components for a real-time resynthesis. The other approach is based on a physically-based simulation to create the sound. We employs the simulation-based approach since the recording of the vortex sound is a difficult task. First, the proposed method creates sound textures for the vortex sound in a pre-process. We use the numerical fluid analysis to cre-

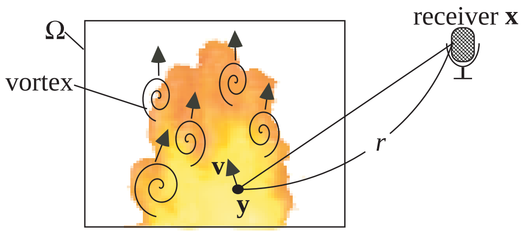

Figure 1: *Principle of vortex sound.*

ate the sound textures. This process can be considered as a virtual recording process of the sound. Next, the sound textures are used for the fast rendering of the vortex sound corresponding to the motion of the fluids. We apply our method to the simulation of the sound of fire. However, the sound of fire consists of many components other than the vortex sound, such as the sound generated by the destruction and deformation of objects due to combustion. Since simulating all of those components is very difficult, we combined synthesized vortex sounds and recorded sounds to enhance the reality of the overall sound. By using the proposed method, we can realize the interactive simulation of vortex sounds corresponding to the motion of fluids.

This paper is organized as follows. First, in Section 2, previous methods that relate to our study are discussed. Next, Section 3 describes the principles of vortex sound. The concepts behind the proposed method are explained in Section 4. Section 5 then proposes a method for the creation of sound textures for the vortex sound. Section 6 proposes a fast method for rendering vortex sound. Examples of the proposed method are demonstrated in Section 7. Finally, Section 8 concludes this paper.

2. Related Work

Research into the generation of sound can be classified into the synthesis of sound (sound modeling) and simulations of the propagation of sound (sound rendering). Since the purpose of this paper is to compute sound waves that represent the vortex sound, let us now discuss previous methods relating to the modeling of sound.

In a pioneering work, Hahn et al. proposed a method for creating sounds procedurally by controlling sound parameters so that the sound changes as a result of changes in the object motion [HGL*95]. In 1998, a method using a sound map for synthesizing contact sound was proposed [vdDP98]. Recently, O'Brien et al. developed methods for computing sound waves by numerical analysis of the subtle oscillation of objects taking into account their shape and material [OCE01][OSG02]. van den Doel et al. used modal resonance models calibrated to recorded sounds in order to simulate the sound of objects colliding, rolling and sliding [vdDKP01]. These methods realize the automatic generation of realistic sound. In these methods, however, the sound of fluids, including vortex sounds, is not taken into account. These methods focus on the sound generated by the oscillation of solid objects.

Dobashi et al. developed a method for the real-time rendering of aerodynamic sound as an example of a method for generating sounds due to the motion of fluids [DYN03]. This method can simulate the sound that is generated when objects are placed in a flow. However, a turbulent field such as fire can generate sound even if no objects are placed in the fluids. Therefore, we cannot apply the method to the simulation of vortex sounds. An alternative approach to synthesizing sound from a turbulent field is to use recorded sound. Some methods synthesize sound waves from the recorded sound waves [DBJR*02][PS03]. The characteristics of the synthesized sound are very similar to the recorded sound. However, this approach cannot always create sound corresponding to the motion of fluids.

On the other hand, researchers in the field of computational fluid dynamics have developed methods for predicting the sound generated by fluids [Tam95][Lel97][How03]. The purpose of this research is to reduce the noise due to high-speed transportation facilities and air-conditioners etc. However, a very complicated fluid simulation is required in this field in order to predict the sound precisely. This results in a large computational cost. Therefore, this approach is not appropriate for real-time applications such as VR.

The method proposed in this paper realizes the rendering of vortex sounds almost in real-time by making use of the predictive methods developed in the field of computation fluid dynamics.

3. Theory of Vortex Sound

In a turbulent field such as a fire or a jet stream, sound is often generated due to the complex motion of fluids. As shown in Fig. 1, the main cause of this type of sound is considered to be the complex motion of the vortices that are generated in fluids [How03]. Experimentally, the amplitude and frequency of the vortex sound is known to depend on the average speed of the turbulent flow [How03]. Lighthill established a theory that relates the fluid motion to the resulting sound [Lig52]. This theory combines wave equations with Navier-Stokes equations to explain the source of sound generated in fluids. Powell and Howe proved that the source of the sound is approximately represented by the term $\rho_0 \mathrm{div}(\mathbf{w} \times \mathbf{v})$ [Pow64][How03], where ρ_0 is the standard sound pressure, $\mathbf{w}$ is the vorticity vector and $\mathbf{v}$ is the velocity vector. By using this approximation and under an assumption that the size of the turbulent field is sufficiently small relative to the wavelength of the sound, they solved the wave equation and derived an analytical expression for the sound pressure p at time t given by the following equations [How03].

$$p(t) = \frac{\rho_0 c_0^2}{12\pi r^3} \frac{\partial^3}{\partial t^3} \int_\Omega f(\mathbf{x}(t), \mathbf{y}) d\mathbf{y}, \quad (1)$$

$$f(\mathbf{x}(t), \mathbf{y}) = (\mathbf{x}(t) \cdot \mathbf{y})((\mathbf{y} \times \mathbf{w}(\mathbf{y}, t - r/c_0)) \cdot \mathbf{x}(t)), \quad (2)$$

where c_0 is the speed of sound, $\mathbf{x}$ is the position of the receiver, Ω represents the turbulent field, $\mathbf{y}$ is a point in Ω, and r is the distance between $\mathbf{y}$ and the receiver (see Fig. 1). Since this model allows the movement of the receiver, its position $\mathbf{x}$ is expressed as a function of t. The origin for $\mathbf{x}$ and $\mathbf{y}$ is placed inside Ω (e.g., the center of Ω). As shown in Eq. 1, the sound pressure is represented as the third derivative of a function related to the vorticity $\mathbf{w}$. The vorticity distribution can be computed from the velocity distribution. Therefore, the sound pressure at the receiver is obtained if we can compute the velocity distribution by using numerical fluid analysis.

4. Basic Concepts of Our Method

Fig. 2 shows the concepts of the proposed method. As shown in Fig. 2(a), the turbulent field is simulated by using $N_x \times N_y \times N_z$ voxels and the proposed method generates sound corresponding to the motion of fluids obtained by numerical fluid analysis. The basic idea is to compute the vorticity distribution by fluid analysis and to calculate Eq. 1 in order to generate the vortex sound. However, in order to use Eq. 1, the size of the turbulent field must be sufficiently small relative to the wavelength of the sound. Therefore, as indicated by the thick lines in Fig. 2(a), the analysis volume is subdivided into smaller sub-volumes. The vortex sound is computed approximately by applying Eq. 1 to each of the sub-volumes and by summing the resulting sound from each sub-volume. This is equivalent to assuming that there are independent virtual point sound sources at the centers of each sub-volume (Fig. 2(a)). In the following, we call the sub-volume corresponding to each point sound source a *sound source volume*. The above approach, however, cannot generate sound at interactive rates, since the time step in the fluid analysis must be sufficiently short to capture the period of the sound (generally, from 10^{-4} to 10^{-2}[s]). Furthermore, a large number of voxels are required to simulate the complex structure of the turbulent field. This results in a long computation time. To address this problem, sound textures for the vortex sound are created in a pre-process. Then the sound is generated at interactive rates using the sound textures.

First, in the pre-process for creating the sound textures, we prepare a small analysis volume whose size is equivalent to that of the sound source volume as shown in Fig. 2(a). This volume is subdivided into finer voxels ($n_x \times n_y \times n_z$) and a turbulent flow with an average flow speed u is simulated within this volume. The sound pressure is then computed using Eq. 1. The resulting sound waves are stored together with the average speed u as a 1D sound texture. Each sound texture represents sound generated by the simulated turbulent flow with the average speed. As shown in Fig. 2(b), several sound textures are created by simulating turbulent flows with various average flow speeds. These 1D textures are collected into a single 2D texture. In the following, the sound texture is denoted as $Q(u,t)$ $(0 < u \leq U, 0 < t \leq T)$, where t is time and U and T are the sizes of the texture.

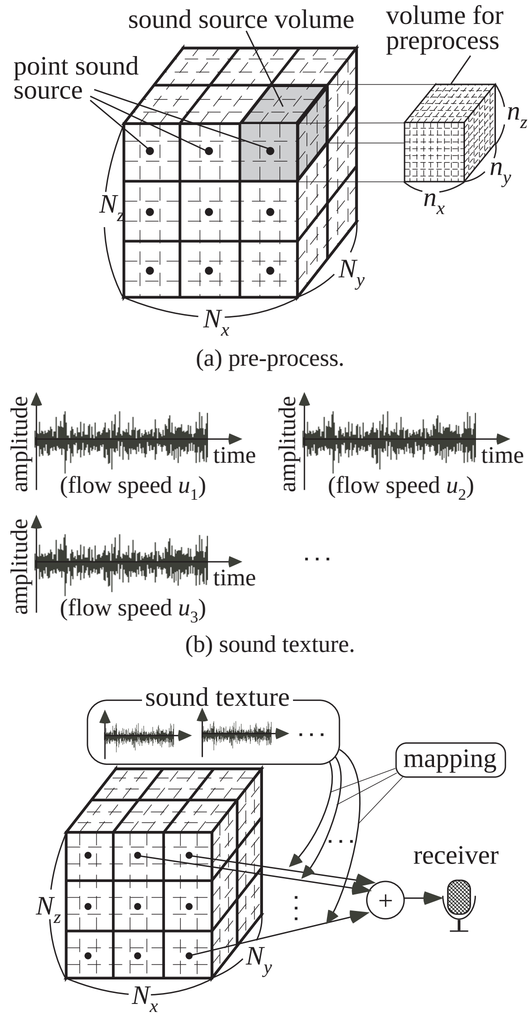

Figure 2: *Basic idea of our method.*

Next, the vortex sound according to the fluid motion is synthesized efficiently by *mapping* the sound textures to the point sound sources. The mapping process is as follows. The motion of fluids is computed within the original volume $N_x \times N_y \times N_z$. Using the result of the fluid analysis, the average flow speed u_i of each sound source volume i is computed. The average speed u_i is used as a texture coordinate to the sound texture. That is, the sound pressure due to each point sound source i is obtained by referring to the sound texture $Q(u_i, t)$. The final sound pressure at the receiver is obtained by summing these pressures (see Fig. 2(c)).

The following sections describe details of the creation of sound textures and the fast rendering of the sounds.

5. Sound Texture for Vortex Sound

As described in the previous section, we simulate the turbulent flows by using an analysis volume (represented by $n_x \times n_y \times n_z$ voxels) whose size is same as that of the sound source volume. We assume that each voxel is a cube with its edge length Δh. Next, the vortex sound is computed using Eq. 1, and the resulting sound waves are stored as the sound texture. The following subsections describe the method for the computation of the vortex sound by using Fig. 3. Note that Fig. 3 shows the process in a two-dimensional representation for simplicity. The same idea can be applied to the three-dimensional case in a straightforward manner.

5.1. Simulation of Turbulent Field

We use a method developed by Fedkiw et al. for the numerical simulation of the turbulent flow [FSJ01]. As shown in Fig. 3(a), we assume that the direction of the flow is perpendicular to the bottom face of the analysis volume (see Section 5.2 about the reason for this assumption). Under this assumption, a natural boundary condition is applied to the top of the analysis volume and a periodic boundary condition is applied to the sides of the volume. For the bottom of the volume, an appropriate velocity is specified to generate a turbulent flow. However, a uniform flow is generated when a uniform velocity is specified at the bottom, so no turbulence exists in this case and no sound is generated. Therefore, the speed of the input flow is perturbed randomly to generate turbulence. The user specifies the average speed u of the input flow. Turbulent flows with various average speeds are simulated by changing u.

At each time step in the simulation, the sound pressure is obtained by computing Eq. 1 using the method described in the next subsection. The sound waves and the speed u are then stored as the sound texture. During the early steps, however, the simulation is numerically unstable. Therefore, computation of the sound pressure begins after the simulation has reached a stable state. The simulation is considered to be stable if the continuity condition of the Navier-Stokes equations is satisfied within a specified accuracy (see [FSJ01] for more details). The time step for the simulation should be sufficiently short to capture the period of the vortex sound. For example, the primary frequency of the sound of a fire is generally low and less than 1000 [Hz]. Therefore, in this case, the time step is set to 1/1000 [s].

5.2. Computation of Sound Texture

The result of the fluid analysis is used to compute Eq. 1 and the resulting sound waves are stored as the sound texture. For this computation, we place a virtual receiver at distance

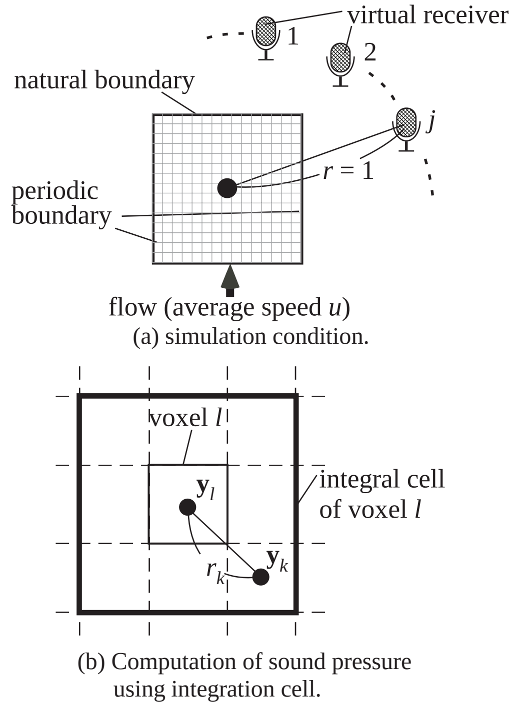

Figure 3: *Computation of sound texture.*

$r = 1$ from the center of the analysis volume. Strictly speaking, the vortex sound has directivity. However, we found experimentally that the directivity is very small and negligible. This implies that the direction of the flow does not affect the resulting sound very much. Therefore, as described previously, the direction of the flow is set to be perpendicular to the bottom face of the analysis volume (see Fig. 3(a)). Multiple virtual receivers are then placed around the analysis volume and the average of the sound pressures at those receivers is stored in the sound texture. That is, at each time step of the fluid simulation, the sound pressures at the virtual receivers are summed and the result is divided by the number of the virtual receivers.

In the computation of Eq. 1, the origin for $\mathbf{x}$ and $\mathbf{y}$ is placed inside the analysis volume. However, since $\mathbf{y}$ is nearly $\mathbf{0}$ near the origin, the function f expressed by Eq. 2 is also nearly zero. This implies that the sound due to the motion of the fluids around the origin is not taken into account. Furthermore, the positional vector $\mathbf{x}$ of the receiver changes depending on where the origin is placed and the resulting sound also changes.

To address this problem, we propose a computation method using *integral cells* for the sound pressure. An integral cell is assigned to each voxel. The integral cell of voxel l consists of voxel l and its neighboring voxels (27 voxels in the three-dimensional case). Fig. 3(b) shows the integral cell in the two-dimensional case. The integral cells of neighboring voxels overlap with each other. The sound pressure due to the motion of fluids inside each integral cell is computed by using Eq. 1. The final sound pressure is obtained by summing the pressures of all integral cells. Let us denote the sound pressure at each virtual receiver j due to the integral cell of voxel l as $\rho_{j,l}$. In the computation of $\rho_{j,l}$, the origin is placed at the center of voxel l. This means that voxel l does not affect the sound pressure $\rho_{j,l}$. However, since the integral cells overlap, the influence of voxel l is taken into account in the computation of the integral cells of the neighboring voxels of voxel l. Therefore, all the voxels are taken into account equally in the computation of the final sound pressure. For each virtual receiver j, the sound pressure $\rho_{j,l}$ due to the integral cell of voxel l is computed by the following equation.

$$\rho_{j,l}(t) = \frac{\rho_0 c_0^2}{12\pi}\frac{\partial^3}{\partial t^3}\sum_k w(r_k) f(\mathbf{x}_j - \mathbf{y}_l, \mathbf{y}_k - \mathbf{y}_l)\Delta h^2, \quad (3)$$

where, $\mathbf{x}_j$ is the position of virtual receiver j, $\mathbf{y}_l$ is the center of voxel l, and $\mathbf{y}_k$ is the center of its neighboring voxel (see Fig. 3(b)). w is a weighting function to take into account the overlap and r_k is the distance between $\mathbf{y}_l$ and $\mathbf{y}_k$. It is reasonable that the weighting function should increase if the distance between $\mathbf{y}_l$ and $\mathbf{y}_k$ becomes short. Therefore, we use the following function as the weighting function w.

$$w(r_k) = \exp(-\alpha r_k) / \sum_{l'=1}^{m} \exp(-\alpha r_{l'}), \quad (4)$$

where α is specified by the user, $r_{l'}$ is the distance from $\mathbf{y}_k$ to voxel l' whose integral cell includes $\mathbf{y}_k$, and m is the number of such voxels. The denominator in Eq. 4 is for normalization. We determined α experimentally and use 3.0 for α in the examples shown in Section 7.

The sound waves obtained by the above method and the average speed of the turbulent flow are stored as the sound texture $Q(u,t)$.

6. Fast Rendering of Vortex Sound

The vortex sound is rendered at interactive rates by using the sound texture $Q(u,t)$ created in the previous section. The motion of the fluids is computed by numerical fluid analysis using the original voxel space of $N_x \times N_y \times N_z$ (see Fig. 1). Again, we use the method developed by Fedkiw et al. for the numerical fluid analysis since the method is very efficient and suitable for interactive simulation [FSJ01]. The sound pressure at the receiver is efficiently computed by just referring to the sound texture. The velocity distribution obtained by the fluid analysis is used to refer to the texture. Details of the method are described below. Let us assume that the process starts at $t = 0$. Computation of the sound pressure is repeated at short intervals of time, Δt. The velocity distribution and the image are updated at a different time interval T_s ($\Delta t < T_s$). In the following, $t_k = k\Delta t$ and $t_j = jT_s$ where k and j are non-negative integers.

Let us consider the case where time $t = t_k$. Firstly, the velocity distribution at time t_k is computed by interpolating the velocity distributions at times t_j and t_{j+1} where $t_j \le t_k < t_{j+1}$. Next, the average velocity u_i of the sound source volume of the point sound source i is computed. Using the sound texture, the sound pressure due to this point sound source is given by $Q(u_i, t_k)$. Since the texture sizes U and T are finite, the texture coordinates (u_i, t_k) are modified as follows. If u_i exceeds U, u_i is truncated to U. If t_k exceeds T, the sound texture is used periodically and the ends of the texture are blended to yield a smooth transition of the sound. Next, the distance r_i between point sound source i and the receiver $\mathbf{q}$ is computed. The sound is delayed due to the distance r_i and attenuated proportionally to $1/r_i^3$ (see Eq. 1). The final sound pressure at the receiver is obtained by summing the sound pressure due to each point sound source, i.e.,

$$p_v(\mathbf{q}, t_k) = \sum_{i=1}^{n} \frac{1}{r_i^3} Q(u_i, t_k - r_i/c_0), \quad (5)$$

where n is the number of point sound sources.

```
renderSoundandImage() {
  Update flow field;
  for(k=0; k<N; k++) { /* N = Ts/Δt */
    ρv[k] = 0.0;
    for(i=0; i<n; i++) {
          /* n: number of point sound sources */
      Compute average flow speed ui inside
       sound source volume i;
      Compute distance ri from point sound
       source i to receiver;
      ρv[k] += Q(ui, t+kΔt)/ri^3;
    }
  }
  Display an image;
  Send(ρv[0], ρv[1], ..., ρv[N-1]) to an audio device;
  t += Ts;
}
```

Figure 4: *Pseudo-code for rendering sound.*

Fig. 4 shows a pseudo-code for rendering the sound using the proposed method. We use a timer callback function for interactive rendering of sounds and images. That is, the function of Fig. 4 is executed in a specified time interval T_s. In Fig. 4, $N = T_s/\Delta t$, p_v is an array for storing the sound pressure at the receiver and t is the time when the function is called. Initially, t is 0.

7. Results

This section shows several examples of sound effects created by the proposed method. In the following examples, images are created by using the method proposed by Dobashi et al [DKY*00]. This method can render fuzzy objects such as fire very fast. We used a desktop PC with a Pentium 4 (3.06

GHz) processor and a NVIDIA Geforce4 as a video card. The size of images is 400×400

First, the proposed method was applied to the simulation of the sound of fire, as shown in Fig. 5. Fig. 5 shows an image from the simulation and the corresponding sound waves. In this example, we combined the vortex sound created by our method with a recorded sound to simulate sound due to combustion of objects. We used a commercial sound library for the recorded sound. The number of voxels in this simulation is $15 \times 15 \times 20$. The temperature of the heat source is randomly perturbed. The sound changes according to the motion of the fire.

Next, Fig. 6 shows an example of rendering the sound from a flamethrower as an example of sound of a jet stream. In this case, the main source of the sound was the vortex sound, so we did not use any recorded sound. The number of voxels in this simulation is $10 \times 20 \times 10$.

Finally, the sound of an explosion was simulated as shown in Fig. 7. To simulate this, we specified velocities radially around the center of the explosion. After the simulation started, the magnitudes of the velocities were decreased rapidly. This creates an explosive sound in the early stages of the simulation (see Fig. 7). After that, vertical flow is caused due to the heat of the explosion. The vertical flow generates the corresponding vortex sound. With this type of sound, we encountered a serious problem. That is, the range of amplitudes of the sound waves is very wide. If we normalize the sound waves, the explosion looks as if it were very far away. This problem is similar to the dynamic range problem in image synthesis [DM01]. Researchers in the audio engineering community have also discussed this problem and several methods were proposed (e.g., [Cut98]). Here, we employ the simplest approach. The sound waves are simply truncated as follows. Most of the microphones have a maximum sound pressure that they can record. We specified the maximum sound pressure and used it to truncate the amplitude. This method can create a realistic sound of an explosion.

The frame rates of the simulations demonstrated in this section are from 10 to 20 [fps]. The pre-process for creating the sound textures took several minutes. Our method can render realistic sound and images almost in real-time.

Please refer to the accompanying video for movies corresponding to the examples shown in this section.

8. Conclusion

In this paper, we have proposed a method for the realistic simulation of sound from a turbulent field. In the method, the sound texture for the vortex sound is created in a preprocess. We proposed a method for creating the sound texture by making use of vortex theory. The sound texture is created by computing the sound pressures due to turbulent fields with various flow speeds. Next, the sound texture is used to render the vortex sound. The proposed method made it possible to render the sounds of turbulent phenomena such as fire at interactive rates. This enables us to achieve an interactive simulation of fluids including sounds.

A few things remain to be achieved in the future. We applied our method to the sounds of fire and explosions. However, in these phenomena, not only vortex sounds but also sounds due to the rapid expansion of heated air are important. In particular, explosions often generate shock waves. Moreover, we used recorded sound for the destruction of objects caused by the combustion. These elements should be simulated to create more realistic sound effects.

References

[Coo02] COOK P. R.: Sound production and modeling. *IEEE CG&A 22*, 4 (2002), 23–27.

[Cut98] CUTMORE N. A. F.: Dynamic range control in a multichannel environment. *Journal of the Audio Engineering Society 46*, 4 (1998), 341–347.

[DBJR*02] DUBNOV S., BAR-JOSEPH Z., RAN E., LISCHINSKI D., WERMAN M.: Synthesizing sound textures through wavelet tree learning. *IEEE CG&A 22*, 4 (2002), 38–48.

[DKY*00] DOBASHI Y., KANEDA K., YAMASHITA H., OKITA T., NISHITA T.: A simple, efficient method for realistic animation of clouds. In *Proc. SIGGRAPH 2000* (2000), pp. 19–28.

[DM01] DEBEVEC P. E., MALIK J.: Recovering high dynamic range radiance maps from photographs. In *Proc. SIGGRAPH'97* (2001), pp. 369–378.

[DYN03] DOBASHI Y., YAMAMOTO T., NISHITA T.: Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. *ACM TOG 22*, 3 (2003), 732–740. (Proc. SIGGRAPH 2003).

[FMC99] FUNKHOUSER T., MIN P., CARLBOM I.: Real-time acoustic modeling for distributed virtual environment. In *Proc. SIGGRAPH'99* (1999), pp. 365–374.

[FSJ01] FEDKIW R., STAM J., JENSEN H. W.: Visual simulation of smoke. In *Proc. SIGGRAPH 2001* (2001), pp. 15–22.

[HGL*95] HAHN J., GEIGEL J., LEE J., GRITZ L., TAKALA T., MISHRA S.: An integrated approach to sound and motion. *J. of Visualization and Computer Animation 6*, 2 (1995), 109–123.

[How03] HOWE M. S.: *The Theory of Vortex Sound*

(Cambridge Texts in Mathematics). Cambridge University Press, 2003.

[Lel97] LELE S. K.: Computational aeroacoustics: A review. In *American Institute of Aeronautics and Astronautics Paper (AIAA Paper)* (1997), pp. 97–0018.

[Lig52] LIGHTHILL M. J.: On sound generated aerodynamically:i. general theory. *Proc. Royal Society London A221* (1952), 564–587.

[LSV*02] LOKKI T., SAVIOJA L., VAANANEN R., HUOPANIEMI J., TAKALA T.: Creating interactive virtual auditory environments. *IEEE CG&A 22*, 4 (2002), 49–57.

[OCE01] O'BRIEN J. F., COOK P. R., ESSL G.: Synthesizing sounds from physically based motion. In *Proc. SIGGRAPH 2001* (2001), pp. 529–536.

[OSG02] O'BRIEN J. F., SHEN C., GATCHALIAN C. M.: Synthesizing sounds from rigid-body simulations. In *Proc. ACM SIGGRAPH 2002 Symposium on Computer Animation* (2002), pp. 175–181.

[Pow64] POWELL A.: Theory of vortex sound. *J. Acoust. Soc. Am. 36* (1964), 177–195.

[PS03] PARKER J. R., SONNY C.: Sound synthesis for the web, games, and virtual reality. In *Proc. SIGGRAPH 2003 Sketches & Application (Conference DVD)* (2003).

[Tam95] TAM C. K. W.: Computational aeroacoustics: Issues and methods. *American Institute of Aeronautics and Astromatics Journal (AIAA Journal) 33*, 10 (1995), 1788–1796.

[TFNC01] TSINGOS N., FUNKHOUSER T., NGAN A., CARLBOM I.: Modeling acoustics in virtual environments using the uniform theory of diffraction. In *Proc. SIGGRAPH 2001* (2001), pp. 545–552.

[TH92] TAKALA T., HAHN J.: Sound rendering. In *Proc. SIGGRAPH'92* (1992), pp. 211–220.

[vdDKP01] VAN DEN DOEL K., KRY P. G., PAI D. K.: Foley automatic: Physically-based sound effects for interactive simulation and animation. In *Proc. SIGGRAPH 2001* (2001), pp. 537–544.

[vdDP98] VAN DEN DOEL K., PAI D. K.: The sounds of physical shapes. *Presence: Teleoperators and Virtual Environments 7*, 4 (1998), 382–395.

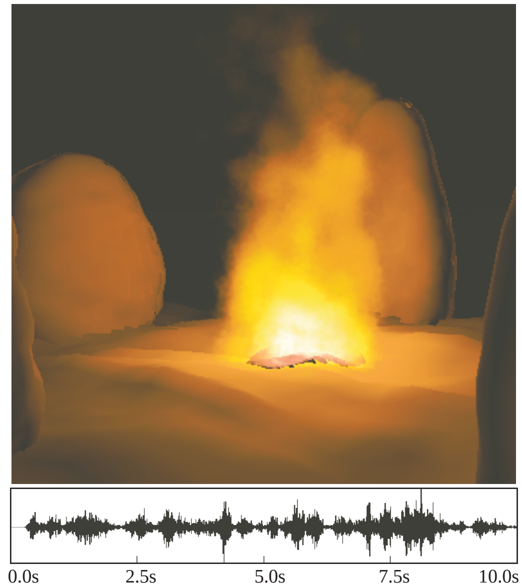

Figure 5: *Sound of fire.*

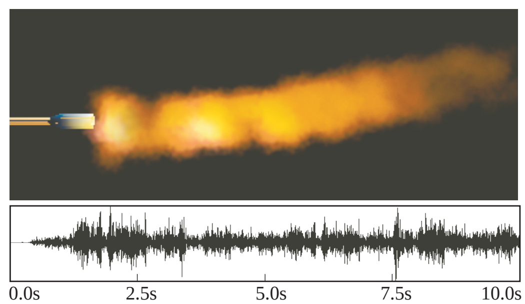

Figure 6: *Sound of flamethrower.*

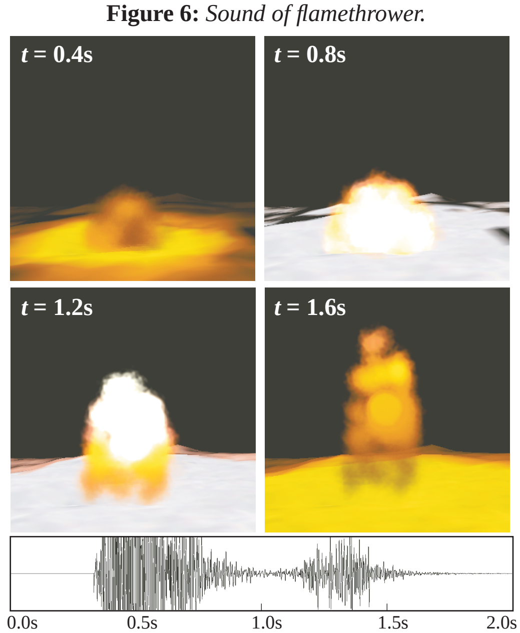

Figure 7: *Sound of explosion.*