Acoustic Modeling of Reverberation using Smoothed Particle Hydrodynamics

Charles Thomas Wolfe
Department of Computer Science
University of Colorado
Colorado Springs, CO
USA, 80933-7150
Charles.t.wolfe@gmail.com

Sudhanshu Kumar Semwal
Department of Computer Science
University of Colorado
Colorado Springs, CO
USA, 80933-7150
semwal@eas.uccs.edu

ABSTRACT

Current methods for digital acoustic modeling of reverberation can be categorized into two main areas: DSP algorithms and sound tracing algorithms. DSP algorithms are usually chosen for their speed and ease of use. DSP methods are generally fast, but do not accurately represent complex environments. Sound tracing algorithms, such as beam tracing, accurately model static environments but have difficulty with complex and/or moving geometry. A Computation Fluid Dynamics (CFD) method called Smoothed Particle Hydrodynamics (SPH) has recently gained interest as a method for accurately simulating fluid flow using discrete particles. This paper extends SPH by adding the ability to model sound wave propagation through fluids. A generalized method that integrates sound generation and reception is presented in this paper. This method provides a basis for acoustic sound effects, such as reverberation.

Keywords

Visualization; Acoustic Field Simulation; smoothed particle hydrodynamics, physical modeling.

1. INTRODUCTION

Reverberation is a natural property of enclosed spaces in which a sound continues to echo after the sound source has been removed. Part of the sound is absorbed and converted to thermal energy, while part of the sound is refracted and continues to travel through the obstacle material. The vast majority of the sound energy, however, is reflected. When a large number of reflections return back to the listener, the effect is called reverberation. The music industry in particular has been interested in artificial reverberation since the early 1900's. With the development of electronic technology, DSP methods are preferred due to their ease of use and easy portability. No longer does a musician or sound engineer has to set up a bulky reverb effect or record in a specialized reverb chamber. The sound is processed electronically after the original sound has been recorded. Other acoustic modeling methods, such as sound tracing, have experienced some success due to their ability to accurately simulate complex environments. Although DSP methods are the current preferred method, they are by no means perfect. The primary restriction on these methods is that they rely on sound engineers recording the reverberation pattern of a given environment. This means for a given reverberation pattern, a room must be constructed, recorded and transformed into a DSP model. The algorithms presented in this paper attempts to eliminate the need for this kind of setup. Using a recently developed fluid mechanics simulation algorithm --- Smoothed Particle Hydrodynamics (SPH) -- complex acoustic environments can be simulated in software. The new algorithms were tested on two sounds -- an acoustic engineering type, the impulse; and a short musical phrase sound.

2. RELATED WORK -- SPH

In Computational Fluid Dynamics, most current methods simulate the flow of fluids using an Eulerian, or grid-based, method [LI64]. SPH, however, is particle-based and therefore utilizes a Lagrangian approach. In an Eulerian simulation, field values are defined only on regularly spaced grid points. The flow of fluid is observed flowing past fixed points in space. This method is well suited for simulation where flow past a point must be observed, such as the end of a pipe. The grid-based method is especially well-suited using data structures such as cellular automata, but the method does come with drawbacks. First, the fluid is confined to the grid and

the flow is undefined outside of the grid. Second, if the grid is large, the data set may only use a small portion of the grid and memory is wasted if the fluid never flows into other portions of the grid. Third, if the fluid particles are to interact with solid objects, the grid resolution may need to be increased to achieve the desired fluid motion. An alternative to the grid-based method above is to use discrete fluid elements, or particles, to represent the fluid [LUCY77]. This gives the benefit of only having to calculate interactions between a finite set of elements and no memory space is wasted on storing field data in locations the fluid may never visit. One downside to this approach is the inherent error in creating a vector field from a weighted contribution of a finite set of particles. Increasing the number of particles can reduce the error, but this also increases the computation time and memory requirements. SPH was created independently in 1977 by Lucy and Monaghan to model astrophysical phenomena [MONA77]. However, the method is general enough to apply to any area of fluid dynamics and has been also used to simulate highly deformable elastic bodies [DESB96]. The central concept of SPH is the use of a smoothing kernel, a function that "smoothes out" the field variables of the particles, and increases the spatial extent of the particles. This kernel is applied to the Navier-Stokes momentum equation, the continuity equation and a state equation. Thus, a field variable of a particle is calculated from a weighted contribution of all particles surrounding it. The smoothing length, *h*, determines the spatial extent of the kernel and therefore what particles are included in the contribution. To evaluate the value of a field variable, such as density or pressure in SPH, a smoothed average is calculated according to the following integral:

$$f(\mathbf{r}) \approx \langle f(\mathbf{r}) \rangle = \int f(\mathbf{r}')\, W(\mathbf{r}-\mathbf{r}';h)\, d^3\mathbf{r}' \qquad (1)$$

where $f(\mathbf{r})$ is the field variable; W is the smoothing kernel; $\langle f(\mathbf{r}) \rangle$ is the smoothed average [CASH02].

The smoothing length, *h*, determines the particles that are used for the interpolation. In discrete simulations, the smoothed average of the field variable can be approximated by summations. SPH uses the concept of a Monte Carlo summation, where points are randomly distributed throughout the fluid. Thus, the probability of finding one of the points in the volume element dV centered around the point $\mathbf{r}$ is proportional to $\rho(\mathbf{r})dV$. Then, the smoothed average can be approximated by the summation:

$$\langle f(\mathbf{r}) \rangle \approx \sum_{j=1}^{N} f_j \frac{m_j}{\rho_j} W(\mathbf{r}-\mathbf{r}_j;h) \qquad (2)$$

where m_j is the mass of the j^{th} point, ρ_j is the density at the position of the j^{th} point, f_j is the value of the field variable being approximated at j. Because of the page constraints we are going to eliminate equations from our discussion. These equations can be found in [Wolf07]. One of the advantages of this method is the ease in calculating the spatial derivatives of field variables, such as gradients and Laplacian. To simulate the flow of a fluid, equations that describe the motion of the fluid must be evaluated. The equation that determines the force acting on given volume of incompressible fluid is the Navier-Stokes equation for incompressible fluids. SPH formulations have been defined to calculate the force on a particle *i* due to the pressure gradient. However, the forces between the particles are not symmetrical. To alleviate this problem, the average pressure between the particles is estimated [LIU03]. Equations are also defined to calculate the viscous force using the general viscous term. In addition, another form of the viscous force equation has been suggested by Müller [MÜLL03]. The external forces, such as gravity, do not depend on the SPH formulation and are just added to the pressure and viscous forces. In order to calculate the force equations, the density must be calculated at every time step. Unfortunately, for free surface flows, the density will be incorrectly low for surface particles, since there are no particles beyond the surface to be included in the density calculation. Monaghan suggests a method for density calculations [MONA94]. First, all particles are assigned an initial density (usually 1000 kg/m^3 for water) and the density only changes when the particles are in relative motion. The rate of change of density for a particle can then be calculated. An equation of state is also used to calculate the pressure of each particle during each time step. The simulation of the breaking dam by [ROY95] uses this method.

There are two criteria that smoothing kernels must meet [MONA92]:

$$\int W(\mathbf{r}-\mathbf{r}';h)d\mathbf{r}' = 1 \text{ and } \lim_{h\to 0} W(\mathbf{r}-\mathbf{r}';h) = \delta(\mathbf{r}-\mathbf{r}').$$

This allows many different kernels can be created to satisfy certain conditions needed in a given simulation. By far the most commonly used kernel is the Beta-spline kernel in [MONA92]. Desbrun suggests a "spiky" kernel for use in computing pressure forces, as the Beta-spline kernel above tends to cause particles to cluster since the gradient of the kernel approaches zero as particles move very close together [DESB96]. Using the standard kernel for the viscosity force calculation can cause the simulation to become unstable if low numbers of particles are used, as they are in real-time

applications. To overcome this, Müller has suggested a different kernel for the viscosity force calculation [MÜLL03]. Although the theory behind SPH is straightforward, there are many details to consider for a successful implementation of a general SPH solver, such as fixed and variable smoothing lengths, symmetrization of particle interactions, adaptable time steps and treatment of boundaries. The smoothing length, *h*, has a direct correlation to a successful implementation of SPH. If *h* is too small, there may not be enough neighbors in the support domain to exert forces on the particle of interest. However, if *h* is too large, local properties of the fluid at the particle of interest may be overly smoothed out, resulting in a low accuracy. The ideal number of neighboring particles (including the particle itself) has been determined to be 5, 21 and 57 respectively for 1, 2 and 3 dimensions [LIU03]. There are many ways to adapt the smoothing length to keep the number of neighboring particles relatively constant. According to [LIU03], the simplest way to evolve *h* over the simulation is by using the average and initial density. If the smoothing length is forced to evolve in both time and space, interactions between neighboring particles will not necessarily conform to Newton's Third Law, as each particle will have its own smoothing length. The following situation could exist: particle *i* includes particle *j* in its sphere of influence, but not vice versa (therefore, particle *j* would exert a force on *i*, but not *i* on *j*, in violation of Newton's Third Law). To overcome this inconsistency, several different methods have been introduced, such as averaging the smoothing length of each of the particles, and also taking the geometric mean. Time integration plays an important role in providing both a stable and fast simulation. The time step must be small enough to ensure that the Courant-Friedrichs-Levy (CFL) condition is met; while not so small that the simulation is computationally infeasible. The discrete SPH equations can be integrated using the familiar Leap-Frog, predictor-corrector and Runge-Kutta methods. For each of these methods, the time step must adhere to the CFL conditions above. To ensure this, the time step is modified to be proportional to the smallest spatial particle resolution, which in SPH is represented by the smallest smoothing length:

$$\Delta t = \min\left(\frac{h_i}{c}\right) \qquad (3)$$

where h_i is the smoothing length for particle *i*, and *c* is the speed of sound through the fluid. The treatment of boundaries has been of great interest since the development of SPH. The symmetric nature of the smoothing kernel produces a particle deficiency at the fluid boundary, as there are no particles on the other side of the boundary to include in the density summation. Therefore, several methods exist that address this problem, and the related issue of boundary penetration. There are generally three ways to prohibit boundary penetration by the particles: collision detection and response, force fields and virtual particles. Standard collision detection techniques, such as point-plane intersections work to keep the fluid particles from penetrating the boundary, but they are problematic. For example, if a particle is reflected off of a boundary, the simulation must also perform collision detection against all other boundaries after the collision response. This can lead to costly, sometimes unnecessary computations. The collision detection/response method also does not address the issue of particle deficiency at the boundary. Another method that seems to work better is to use a force field at the boundary [AMA05]. During each time step, each particle in the fluid receives a force component from every boundary. The force equation successfully prevents the particle from penetrating the boundary. The use of stiffness and dampening in the force equation allows for simulation of different boundary materials, such as concrete or foam. The force field method still suffers from particle deficiency at the boundary, but the ability to simulate different materials at the boundary is a useful tradeoff. Monaghan [MON94] suggested a popular solution to this problem. A grid of "virtual particles" is placed on the boundary. The virtual particles have the same density as the initial density of the fluid. As a fluid particle nears the boundary, the virtual particles exert a force on the real particle that resembles the Lennard-Jones potential. The use of stiffness and dampening in the force equation allows for simulation of different boundary materials, such as concrete or foam. The force field method still suffers from particle deficiency at the boundary, but the ability to simulate different materials at the boundary is a useful tradeoff.

3. SPH ACOUSTIC MODELING

Initialization

Initialization of the simulation occurs when an SPH scene file is loaded. This scene file specifies configuration for the following: global settings, fluid parameters, boundary conditions, the sound emitter and the sound receiver. The global settings configuration specifies the simulation time step and any external global forces, such as gravity. The fluid settings specify a cuboid shaped lattice of SPH particles defined by a position, width, height and length. The initial density, equation of state stiffness constant and viscosity constant are also defined here. For each boundary, the position, orientation, size and energy absorption properties are defined. The

boundary can be one of three different shapes: rectangle, disc and cylindrical surface. There can be more than one boundary in the scene file and complex structures such as rooms, halls and cathedrals can be constructed from individual boundary objects. The sound emitter specifies the input WAV file (the source sound), position and size. The emitter is based on a disc type boundary. The sound receiver specifies the output WAV file (the source sound with any reverberation from the environment) and a listening position

Simulation – Stage 1

Before a source sound can be processed through the simulation, the fluid must first reach an equilibrium state. The system is said to be in equilibrium when boundary forces acting on the fluid are equivalent to forces due to the pressure gradient internal to the fluid itself. To reach this state, the simulation must proceed from initialization without any sound source inducing energy into the system. The system runs in this state until a satisfactorily low energy is reached:

$$E_{\text{system}} = \frac{1}{2}\sum_{i=1}^{n} m_i v_i^{\,2} < L \qquad (4)$$

where L is a sufficiently small number (we use 0.00001 J). When equilibrium is reached, the simulation then proceeds to stage 2 below.

Simulation – Stage 2

After equilibrium is reached, the emitter is engaged, which propagates the source sound through the fluid. At the end of each time step, the sound receiver calculates the density at its location. The density at the receiver is compared with the reference density of the fluid. Any excess density will be recorded as a positive deflection in the output sound file, whereas low density will be recorded as a negative deflection.

New Sound Emitter Algorithm

The sound emitter is an object that acts as a boundary in the simulation, but is allowed to move according to a positive or negative deflection from the DC value in the input sound file. A positive deflection will result in a positive movement in the emitter along the x-axis. A negative deflection in the input sound file will result in a negative movement of the emitter along the x-axis. The magnitude of the movement along the x-axis is dependent upon a user-definable deflection factor that is set in the scene file. The magnitude of the x-axis movement is determined by the following equation:

$$x_t = \frac{s_t k}{32768} \qquad (5)$$

where s is the input sample (a 16-bit signed integer), t is the current sample and k is the deflection factor. In effect, this states that the emitter cannot be deflected more than k units on either the positive or negative x-axis. The emitter propagates the sound wave through the fluid by interacting with the fluid particles that are its neighbors. As the emitter moves, the Lennard-Jones potential induces a force on the fluid particles, causing the fluid to move in sync with the emitter, and thus the source sound itself.

New Sound Receiver Algorithm

The purpose of the sound receiver is to capture the vibrations in the fluid as an acoustic wave. When the simulation first enters stage 2, before the sound emitter starts to move, the fluid density is calculated at the receiver location. This density is stored as the receiver's reference density and is used in subsequent time steps to determine the positive or negative deflection from the reference. This deflection is then stored as a sample in the output sound file according to the following equation:

$$s_t = (\rho_0 - \rho_t) \cdot 32768 \qquad (6)$$

where s_t is the t^{th} sample, ρ_0 is the receiver's reference density and ρ_t is the density of the fluid at the receiver's location. The magic number of 32768 is the sample conversion equation represents the maximum deflection of the pressure wave and is directly related to the size of the 16-bit integer, since we are converting fluid density values in the range of -1.0 <= pressure <= 1.0 to 16-bit sound samples. As in the input sound file, the sample is stored as a 16-bit signed integer. To ensure that background noise is filtered out, the output sound file's DC component is determined, the entire sound file is adjusted to make the DC component zero, and then the sound file is normalized.

4. IMPLEMENTATION

Since the SPH approach to acoustic modeling has not been attempted before to the best of our knowledge, a test plan was created to find the variable settings that produced the most desirable results. The simulation has the ability to load individual scene files, which are comprised of the simulation variables and an enclosed space generated by SPH fluid boundaries. The simulation variables that are exposed via the scene file are: timestep and fluid parameters (# of particles, initial volume, mass, viscosity and stiffness constants). In all simulation runs, we ignore temperature changes in the fluid. Acoustic energy is converted to heat by specifying different values for the boundary dampening constant. In addition to the simulation variables, complex environments can be created by assembling individual boundaries (walls) into rooms or halls. For this paper, three different types of boundaries

were created: rectangular, cylindrical and disc-shaped. In all test environments generated, a rectangular or box-shaped room was assembled from six rectangular boundaries. For a boundary, the size, orientation, stiffness and dampening can be adjusted to simulate materials with different acoustic properties, such as sound absorption (low stiffness, high dampening) or sound reflection (high stiffness, low dampening). The sound emitter can be of any size, but is restricted to a disc-shaped object. Since the emitter is also a boundary, stiffness and dampening can also be specified, but is usually set to high stiffness and low dampening so the sound can be propagated through the system efficiently. A deflection constant can also be specified for the emitter, guaranteeing that the emitter will never move beyond a certain distance from its rest position. The sound receiver specifies a single output WAV file and a world position vector.

The system uses C++ with performance in mind. To visualize the particle system in 3D, OpenGL was employed as the graphics engine. The SPH acoustic simulation developed for this paper is based on the core of a water splash simulation that was written by the first author. For the water splash animation, an SPH solver simulated water as a compressible fluid and used as an artificial equation of state to keep the fluid particles at a somewhat constant density. For the acoustic simulation, the bulk of the SPH calculations remained the same, with the equation of state for an ideal gas being substituted in place of the artificial equation of state for water. The following processing methods are provided by SPHSimulator class --

(a) Load() – Reads in a scene file and sets up the system to simulate the environment specified in the file. Particle creation and initialization is done during this step.

(b)StepSimulation – Steps the simulation forward one time step (the duration of the time step is defined in the scene file). The following pseudocode outlines this process:

(b1) For each particle, assign neighboring particles to neighbor list.

(b2) For each particle, compute its density according to Equation 11 in [Wolf07].

(b3) For each particle, compute its pressure using Equation 27 in [Wolf07].

(b4) For each particle, compute the force acting on the particle due to the pressure gradient (Equation 8 in [Wolf07] and the viscous force (Equation 28 in [Wolf07]. Integrate the equations of motion to find the new velocity and position according to the Leap-Frog method.

(b5) For each sound emitter, read the next sample in the WAV file and move the emitter accordingly (see SPHEmitter section below).

Class SPHKernel and Derivatives

The SPHKernel class is an abstract class the enables the simulation to use drop-in kernel equations. Three different kernel types are supported: Monaghan's B-spline kernel, Desbrun's spiky kernel and Müller's viscosity kernel.

Classes SPHGrid and SPHGridCell

When the SPHSimulator class performs an AssignNeighbors() action, a grid is superimposed on the particle collection. This grid is used as a spatial data structure, with cubic buckets having length *h*. The buckets store references to SPHParticle objects, which are used to perform a nearest-neighbor search algorithm. After the AssignNeighbors() action is performed, each particle will have a list of its neighbors that it uses for the rest of the SPH computations for the current time step.

SPHSoundEmitter

The sound emitter class is derived from one of the concrete classes based on SPHBoundary. In effect, it is a boundary that moves according to input samples from a WAV sound file. The emitter is centered on a rest position, and then vibrates according to the sound file defined for the emitter. A configurable maximum deflection is used to provide amplitude control for the produced sound wave. For each time step that is processed, the emitter reads the sample corresponding to the time step, and moves itself along its local x-axis by an amount defined by:

$$x = x_{\text{rest}} + (ks_t) \tag{7}$$

where x_{rest} is the rest position, k is the maximum deflection and s_t is the sample at time t such that $-1 \le s_t \le 1$.

SPHSoundReceiver

The SPHSoundReceiver class simulates a sound receiver, such as a microphone. When the SPH simulation starts, the receiver calculates the density of the fluid at its location based upon the weighted sum of the particle masses surrounding it It stores this initial value as a reference density. For each time step afterwards, the density is again measured by the same method and compared against the reference density. This deflection (negative or positive) is stored as a double-precision floating point value between -1 and 1. In addition, an output WAV file is created at the end of each time step that represents the sound wave the receiver detected up to and including that time step.

Class Camera

To visualize the particle system, a camera with the ability to move in all three dimensions as well as rotate around all three axes was created. A Camera object is created for each OpenGL window used in the simulation (currently only one view is used). A reference to this camera is then passed into the SPHSimulator, and is then passed again to the SPHParticle and SPHBoundary classes during the rendering phase.

5. SIMULATION RESULTS

For the simulation runs, two different WAV files were created. The first was designed to produce a very short impulse and very quickly decay to zero after the burst (Figures 1-2). The impulse is an attempt to recreate the traditional method of determining an environment's reverberation properties, such as firing a pistol filled with blanks. The second WAV file is a music application of the simulation and consists of a brief Latin piano melody with a shaker accompaniment (Figures 3-4). To compare the output from the simulator to a professional reverberation effect, an additional WAV file was created by processing the piano sample with the "Hall 1" reverberation effect from a Yamaha S90 ES synthesizer and recording the result. In Figures 1 through 4, the time domain and frequency domain graphs for both the impulse and piano melody are displayed.

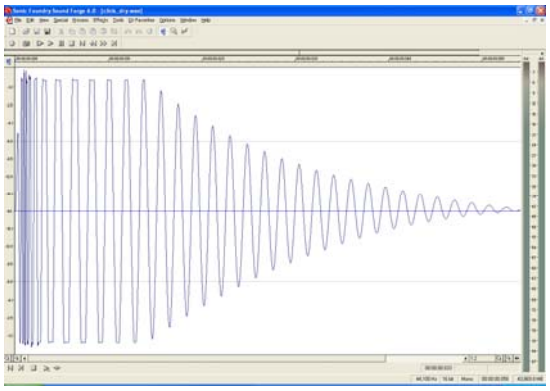

Figure 1: Time and domain graph of impulse.

Simulation time step	2.268×10^{-5} s
Fluid stiffness constant	119.215
Fluid viscosity constant	1.0×10^{-4}
Smoothing length	2.130×10^{-2} m
Mass per particle	1.935×10^{-6} kg
Particle rest density	1.29 kg/m^3
Environment volume	5.488×10^{-2} m^3
Boundary stiffness constant	1.0×10^{9}
Boundary dampening constant	0.1
Emitter radius	2.5×10^{-2} m
Emitter stiffness constant	3.0×10^{8}
Emitter dampening constant	0.1
Emitter deflection factor	0.001 m

Table 1: Samples of values used in simulation.

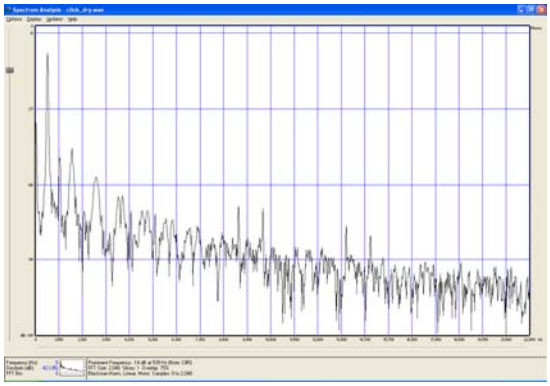

Figure 2: Frequency domain graph of impulse.

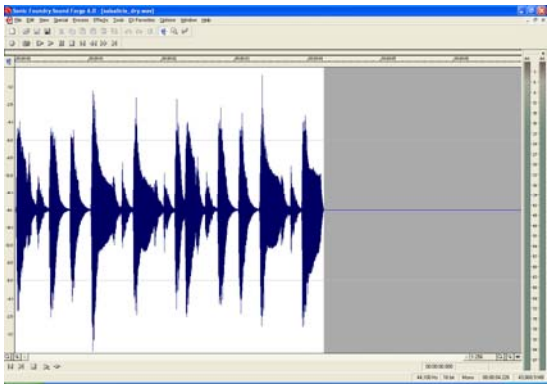

Figure 3: Time domain graph of piano melody.

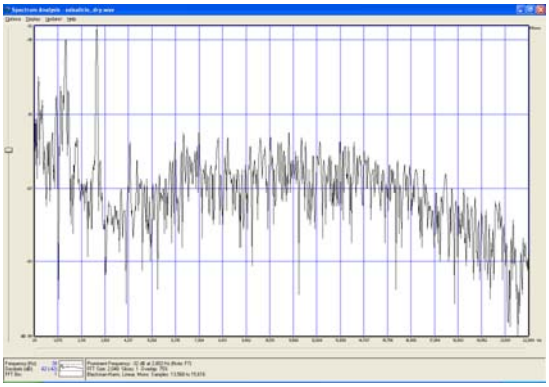

Figure 4: Frequency domain graph of piano melody.

The first simulation scene was created by inputting a "best guess" value for each parameter in the simulation. The time step is set at 1/44100, or the sampling frequency of the input WAV file. The fluid stiffness constant was generated by taking the Ideal Gas Equation and solving it for the appropriate constant *k* for air for a known pressure and density. The viscosity constant was set as low as possible without sacrificing stability in the SPH equations for the specified time step duration. The boundary stiffness and dampening were set so as to generate a high reflection factor with minimal acoustic energy loss. The emitter size and deflection factor were set to appropriate values to simulate a loudspeaker in the environment (Table 1, more details in [Wolf07]). Although the simulation did produce reverberation, the frequency spectrum in Figure 6 shows that the output impulse had a power gain around 4000 Hz. This data agrees with the author's observation of hearing the reverberation as a hollow, metallic sound. The second simulation run using scene file #1 used the piano melody as the input WAV file. As with the impulse WAV, a hollow, metallic reverberation was observed. One interesting artifact with this simulation is observed around 5500 Hz: due to the smoothing effect of the SPH interpolation, the sound was effectively low-pass filtered starting around 5500 Hz. This produced a muddy sound quality in the output WAV. Using more fluid particles may produce better results, but the resulting computation time would be too great to be of any great value. We performed several (six) similar experiments with different scene files which have been omitted due to the page size constraints, but can be found in [Wolf07].

The results from the simulation of scene file #1 show promise. Using the impulse WAV as input to the simulation, the resulting output is similar in sound. Reverberation can be heard as the impulse decays, and has a hollow, metallic quality. In the time domain graph shown in Figure 3, the sound receiver's initial density is incorrectly high, most likely because of minor fluctuations in density at the receiver position due to the chaotic movement of the fluid particles.

6. FUTURE DIRECTIONS

Before further application possibilities are considered, the accuracy of the reverberation results could be improved. The equation of state is the most obvious place to start to improve the simulation. In this paper, the equation of state used was based on the Ideal Gas Law. The stiffness constant that was required bordered on being too high for the simulation to remain stable at reasonable timesteps. In addition, we did not simulate the thermal properties of the fluid, as we assumed the temperature would be more or less constant and any thermal conversion of the acoustic energy would be simulated in the boundary dampening factor.

Another possibility for improvement is using different smoothing kernels for the density, pressure gradient and viscosity equations. Smoothing kernels have been known to produce odd behavior if not properly adapted to the application [MÜLL03]. Liu and Liu have devoted an entire chapter of their SPH book to constructing custom smoothing kernels [LIU03]. For the pressure gradient and viscosity equations, care must be taken to ensure that the kernel derivative and Laplacian exist.

Although this paper focused solely on simulation of reverberation, another possible application of the simulation, once properly adapted, is to perform general acoustic modeling. This type of simulation is useful in many industries, including building design, automobile, aircraft and marine design and music production. To this end, an equation of state that includes thermal energy must be used, as well as the simulation of the acoustic absorption and refraction effects of different materials. For the equation of state, the standard Ideal Gas Law would still apply for simulation of air, if simulated in the well known $PV = nRT$ form. However, a new boundary design would have to be considered that allowed the sound to propagate through the boundary material. This could be accomplished by treating the walls and other obstacles not as boundaries, but as solid masses of particles that the fluid interacts with. The particles could be connected by a damped mass-spring system that would allow sound wave propagation and thermal energy transfer. With this type of a system, the simulation could easily handle a very dynamic environment, where the boundaries and other obstacles are able to move in time.

7. CONCLUSION

This paper has presented the history and theory of acoustic modeling of reverberation, as well as the history of the Smoothed Particle Hydrodynamics method of fluid mechanics simulation. The lack of a general solution to the acoustic modeling problem when dealing with complex environments was a drawback of earlier methods. New algorithms were developed for sound emission and reception. Using the SPH fluid mechanics algorithm as a base, An SPH simulation application utilizing C++ and OpenGL was developed that allowed the testing of the new algorithms. We have provided a first step to a more robust solution which opens door towards further research using our formulation.

The system as implemented is still far from being a solution to the general acoustic modeling problem. Still our simulation results simulate the reverberation to a small degree of accuracy. This shows that the SPH algorithm is a tool for acoustic modeling simulation of complex environments. When a faster solution is obtained, the use of SPH as a tool in acoustic modeling would prove valuable for music studio design, architecture and acoustic dampening for the automotive, and applications for aircraft and marine industries.

8. REFERENCES

[Cash02] Cash, J. 2002. Modeling the accretion stream in polars. Ph.D. Dissertation, Department of Physics and Astronomy, University of Wyoming.

[Desb93] Desbrun, M. and Gascuel, M. 1996. Xue, L. 2004. A new paradigm for animating highly deformable bodies. Computer Animation and Simulation '96. 61-76.

[Ever01] Everest, F., Master handbook of acoustics, Fourth edition, McGraw Hill, 2001.

[Funk98] Funkhouser, T., et al. 1998. A beam tracing approach to acoustic modeling for interactive virtual environments. *Computer Graphics (SIGGRAPH '98)* 21-3.

[Li64] Li, W and Lam S., Principles of fluid mechanics, Addison-Wesley, 1964.

[Liu03] Liu G and Liu M, Smoothed particle hydrodynamics: a meshfree particle method. World Scientific, 2003.

[Lucy77] Lucy L, 1977. A numerical approach to testing of the fission. The astronomical Journal, Vol 82, no. 12.Li, W and Lam S., Principles of fluid mechanics, Addison-Wesley, 1964.

[Mona77] Monaghan J. Gingold R, 1977. Smoothed Particle Hydrodynamics theory and applications to non=spherical stars, Montly notices of the Royal Astronomical Society 181(2):375-389.

[Mona92] Monaghan J 1992. Smoothed Particle Hydrodynamics. Review of Astronomy and Astrophysics, 543-574.

[Mona94] Monaghan J et a, 1994, Simulation of free surface flows with SPH. ASME Sympsium on Computation Methods in Fluid Dynamics.

[Mull03] Muller M et al 2003. Particle based fluid simulation for interactive applications. Eurographics/SigGraph Symposium on Computer Animation.

[Roy 95] Roy T. 1995. Physically based fluid modeling using smoothed particle hydrodynamics. MS University of Illinois at Chicago, USA.

[Whit01] White P. 2001. Advanced Reverberation Part 1. Sound on Sound, October 2001.

[Whit01] White P. 2001. Advanced Reverberation Part 2. Sound on Sound, November 2001.

[Wolf07] Charles Thomas Wolfe, Acoustic Modeling of reverberation using smoothed particle hydrodynamics, MS Thesis, University of Colorado, Colorado Springs, CO, pp.1-63, 2007.

.