3D PARTICLE SYSTEMS FOR AUDIO APPLICATIONS

Nuno Fonseca
CIIC/ESTG,
Polytechnic Institute of Leiria
Leiria, Portugal
nuno.fonseca@ipleiria.pt

ABSTRACT

Although particle systems are well know for their use in computer graphics, their application in sound is very rare or almost non-existent.

This paper presents a conceptual model for the use of particle systems in audio applications, using a full rendering system with virtual microphones: several virtual particles are spread over a virtual 3D space, where each particle reproduces one of the available audio streams (or a modified version), and the overall sound is captured by virtual microphones.

Such system can be used on several audio-related areas like sound design, 3D mixing, reverb/impulse response design, granular synthesis, audio up-mixing, and impulse response up-mixing.

1. INTRODUCTION

A particle system is a stochastic system made of many virtual small particles that evolve over time. This concept is used extensively on computer graphics [1], as a means to obtain many visual effects that are difficult to create otherwise, like rain, smoke or fire [2], but its use in audio applications is quite rare.

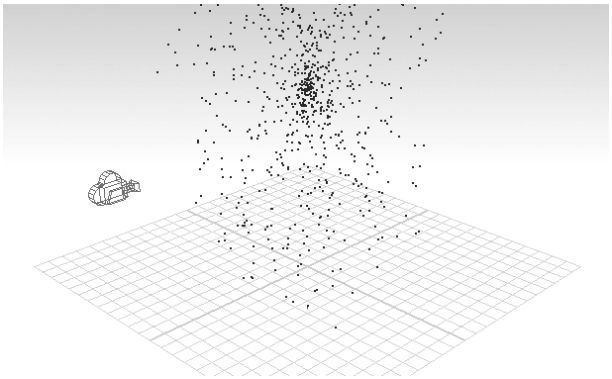

Figure 1: In computer graphics, particle systems are captured with a virtual camera. By replacing the virtual camera with a virtual microphone, the same concepts can be used for audio applications.

In [3], the particle system features of Jitter, the graphic engine of Max/MSP, are used to control a granular synthesizer. The coordinates of the particles were extracted and used to control parameters of the granular voices, including their panning position in a 4-speaker setup with an additional speaker above the listener's head. Later, the same author presents in [4] a different system that uses particle systems to control the space location of spectral components of sound, merging spectral with spatial processing.

The concept of a particle system is used in [5] for synthesizing walking sounds. The system analyses walking sounds (or similar ones), extracts sound related features (gait, heel-toe events, etc.) and tries to resynthesize the walking sound using the concept of a particle resynthesis model.

In [6], the concept of particle is presented, as a way to model diffusion on reverb models, but the concept of a particle system is not mentioned.

More recently, in [7] a particle-based synthesizer for interactive virtual environments is presented. A particle system is created to synthesize rain, fire and wind, using physical modelling approaches based on atoms (particles), that can work in real-time situations (e.g. computer games) to produce sound effects.

This paper presents a conceptual model for the use of particle systems in audio applications. By using a fully rendering system, with virtual microphones, the overall sound of a virtual space, created by a huge number of virtual particles, is obtained. This concept can be applied in several fields of audio, like sound design, synthesis, 3D mixing, and reverb design, among others.

The work presented here is based on a complex audio particle system implemented by the author.

The paper is organized with the following structure: section 2 will present the main concepts of a sound related particle system; section 3 will focus on the rendering process; section 4 presents an overview of several audio applications; and finally, some conclusions and future work are presented.

2. PARTICLE SYSTEMS FOR AUDIO

The goal of this paper is to present a conceptual model that uses particle systems within an audio context, i.e., a system that uses concepts like particles, particle systems, and virtual microphones to render one or more audio streams - each particle will reproduce one of the available input audio streams (or a modified version of it) as they move (or not) through a virtual 3D space where one or more virtual microphones capture the overall sound, which will result in one or more output audio streams. Such a system can be used in many different scenarios and for different applications within audio: as a sound design tool, as a 3D mixer/processor, as a reverb design tool, as an impulse response design tool, as a granular synthesizer, as an up-mixing processor, among other applications.

2.1. Particle System

A particle is a virtual entity with a very basic behavior – a point in space that reproduces sound. Each particle will have a lifetime: they are created, they reproduce their designated audio,

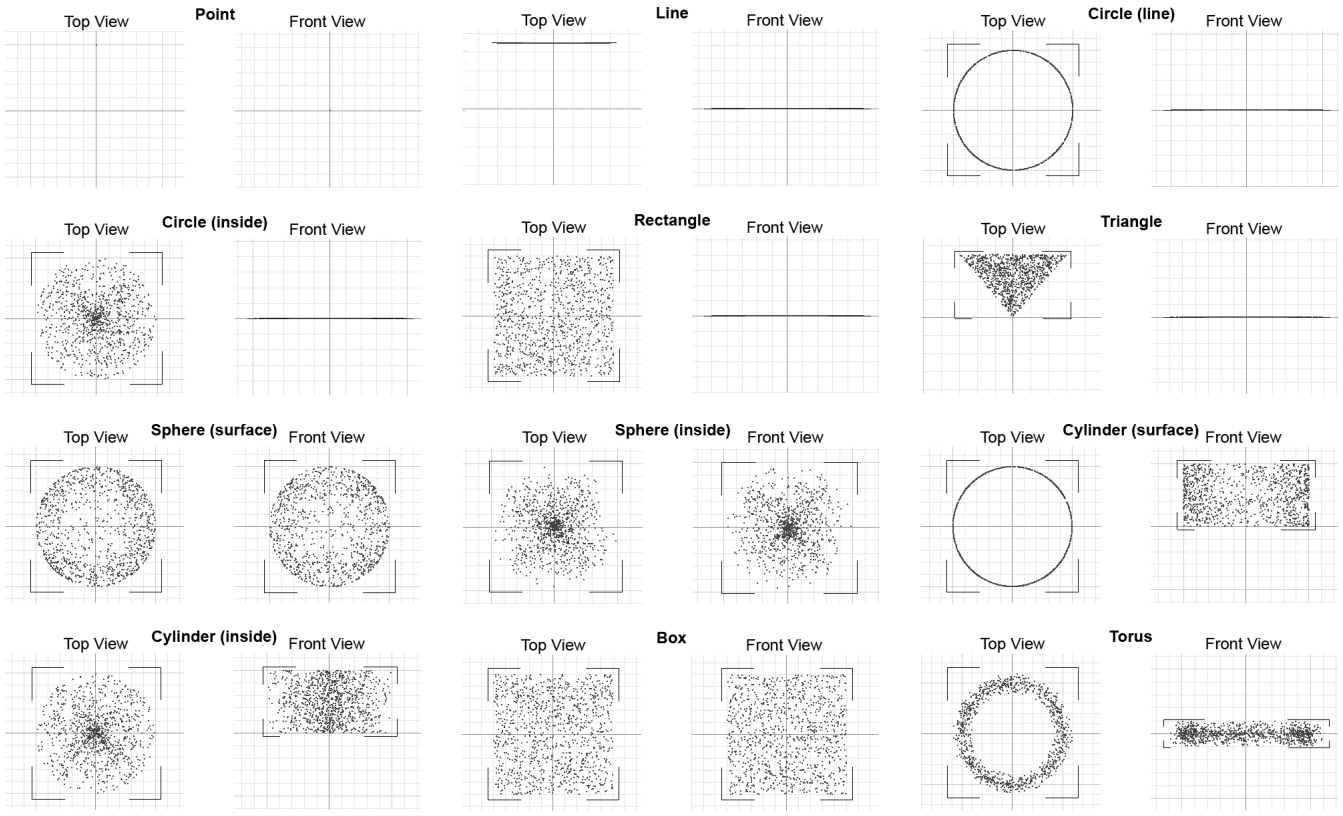

Figure 2: Examples of starting particle zones (images with top and front views, 1000 particles: point, line, circle (line), circle (inside), Rectangle, Triangle, Sphere (surface), Sphere (inside), Cylinder (surface), Cylinder (inside), Box, Torus.

and they die. Also, each particle will have a position, defined by 3 coordinates (x, y, z), that can be static or that can move during its lifetime.

To simulate highly complex situations, a huge number or particles must be generated (e.g. thousands or even millions), and that is the mission of a particle system - the particle system is an entity that creates particles and controls their behavior. One of the things that characterize particle systems in general is that the user does not control each particle individually. The user will control the parameters of the particle system, and based on that information, the particle system will control each one of the particles. For instance, the user may specify that the particle system should create particles on a cube with a width of 20 meters, and then the particle system will create all particles, randomly, but inside the defined cube. The user may also define how a particle property should be chosen, usually in the form of something like (1) (the random function may have a uniform distribution or any other type of distribution).

$$property = centralValue + random() * variance \quad (1)$$

For instance, the user may state that the particles will have an initial delay that ranges uniformly from 0 to 10 seconds, or that the particle will have an original velocity of $[V_x, 0, V_z]$, with V_x ranging from -10 to 10 m/s with a normal distribution, and V_z with a static value of -1.

Each particle system will have one or more audio streams as their sound sources. If there is only one audio stream within the particle system, then all particles will reproduce that audio stream. If there is more than one audio stream, than each particle will randomly choose one of the available audio streams and reproduce only that stream. Audio streams with different durations are perfectly acceptable.

Figure 2 shows some examples of starting particle zones that could be implemented in an audio particle system software.

2.2. Virtual Microphones

The same way that virtual cameras capture the image of particle systems on computer graphics applications, virtual microphones are used to capture the sound of particles on audio applications.

One of the most important parameters of a virtual microphone is its directivity (polar response), and as expected, traditional types like omnidirectional, cardioid, figure-of-eight can be easily implemented.

Since virtual microphones are not real, they can even have characteristics or behaviors that are difficult or even impossible for real microphones to have. For instance, several "virtual" microphones can be located at the exact same location spot, without artifacts due to small distances between capsules. One of the characteristics that can be of extreme importance in this type of applications is custom directivity behavior: highly directional response (e.g., capturing sound only on a frontal 1º angle), only first lobe capture, or asymmetrical directivity response (e.g. a 5.1 microphone setup where the left microphone presents a narrower

polar response to its right, regarding the center mic, but a wider polar response to the left, regarding the left surround mic).

Since different virtual microphones can be used, the same scene can be rendered with multiple microphone setups: stereo, 5.1, Ambisonics, binaural (using HRTF), *n*-equally spaced microphone setup, etc. (see Table 1 for some examples of virtual microphone types). For instance, a contemporary electroacoustic piece can be rendered differently, depending on the speaker setups of the place where its going to be reproduced, ranging from mono to a large and complex custom speaker room (e.g. a venue with 100 independent speakers).

The distance between the particle and the microphone will affect the gain at which the particle sound is rendered (due to the attenuation of the inverse-square law), so there is a special situation that needs to be considered – what should happen if the particle is located at the exact same position as the microphone or near enough? This is a situation that would create a gain so high that would interfere with the dynamic range of the system, since this particle would be rendered with amplitude so high that any other sounds would be outside the human hearing dynamic range. To prevent such situation, each virtual microphone should have a minimal distance parameter – each particle that is located at a distance smaller than this "minimal distance", should use the "minimal distance" value instead. For instance, stating a minimal distance of 10 cm (i.e., no additional gain is applied to particles located at less than 10 cm away from the microphone) is enough to prevent such problems.

Table 1: Some e*xamples of virtual microphone setups.*

Microphone Type
Mono – Omni
Mono – Cardioid
Mono – Hyper cardioid
Mono – Figure-of-Eight
Mono - Custom
Stereo – MS
Stereo – XY
Stereo - AB
Stereo - Blumlein
Stereo – ORTF
Stereo – Binaural (HRTF)
Array – Double MS
Array – LCRS
Array – 5.1
Array – 6.1
Array – 7.1
Ambisonics – 1st order
Ambisonics – 2nd order

2.3. Audio streams and audio modifiers

Although each particle can reproduce the exact same audio stream, in most cases it would be much more interesting to have different audio streams within the particle system or having audio modifiers that make small (or even big) changes on the audio reproduced by each particle.

If several audio streams are available to the particle system, the system randomly allocates one of those streams to each particle (some particles would reproduce audio stream 1, other particles would reproduce audio stream 2, and so on). For instance, in a situation where the user is trying to recreate the sound of a war battle, some particles could reproduce bullet sounds, other particles could reproduce explosions, other particles could reproduce human screams, etc.

Besides using different audio streams, each particle can also apply special audio modifiers to the audio stream that is assigned to it. These modifiers can range from simple random gain modifiers, to random filters (e.g. a FIR filter with random coefficients), band-pass filters/filter banks, pitch/time shifters, grain (time) windows, etc.

In the case of filter-based modifiers, with a high number of particles, statistical information can be taken into considerations. For instance, an audio modifier with an FIR filter with random coefficients (first coefficient of 1 and additional coefficients ranging from -1 to 1), will apply random filter responses to each particle, but from a statistical point of view, the overall frequency response of the system will be similar to the original audio stream, since the mean value of FIR coefficients will be 1 for the first coefficient, and a 0 for the additional coefficients (Figure 2).

In the case of filter bank modifiers, each particle will reproduce essentially only a part of the spectrum. For instance, in a 1/3 octave filter bank, each particle will reproduce only one of the 31 audio bands. The final overall equalization of the particle system can even be changed, not by applying a fixed equalization to each particle or the initial audio stream, but by using a system with a large number of particles and changing the random distribution of the particles (e.g. increasing the number of particles that reproduced a specific band, will increase that frequency range on the final rendered audio).

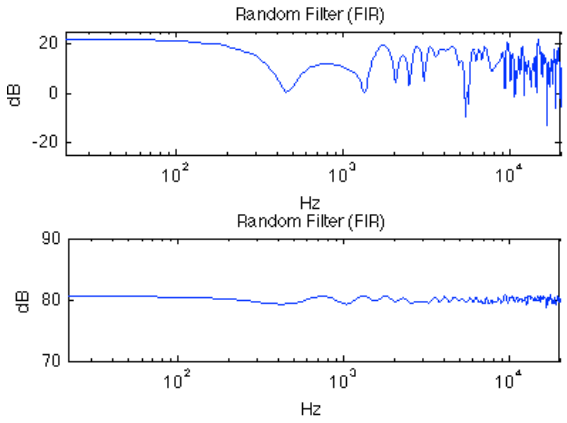

Figure 2: On top, the frequency response of an FIR filter with 100 random coefficients (uniform, [-1,1]). At the bottom, the frequency response of 10.000 such filters combined.

2.4. Position modifiers

It is possible to apply movement to the particles. Although in some situations the user may want static particles - particles that are fixed to their original position - in many applications we may want moving particles, either with a common behavior (e.g. gravity) or even with completely independent behavior (e.g. each particle with its own random direction and random velocity or acceleration).

In some circumstances, circle-based movements may be desired, as a way to get movement but without changing the distance between particles and microphones, preventing time/delay effects.

The position modifiers can also be applied to the zone where particles are created - particles that have an initial delay of *x* may start in a different zone than particles created at instant *y*. The starting position modifier only changes the zone where particles are created, but after the particle is created, only the particle position modifier will affect their future location.

Eventually, the concept of particle collision can also be considered – particles that collide with other particles, changing their movement, with or without generating sound events. In that scenario, two additional parameters could be considered: a particle radius and a particle mass. Since particles are considered as having a near-zero size (volume), the probability of a collision is near zero – a collision exists only if two particles have the exact same position at the same time instant, which would be very difficult within a continuous 3D space with random distributions. By adding a particle radius, a collision occurs when the distance between 2 particles are less or equal to the sum of their radius. Once again, the user may choose to have a fixed radius for all particles or not. The concept of a particle mass, with different values for each particle, would allow different behaviors after a collision – the particle with the lower mass value would suffer a higher change in direction.

3. RENDERING PROCESS

To obtain the final audio streams, the system must perform a rendering process (Algorithm 1), where each virtual microphone will capture the sound of each particle.

Algorithm 1: *Rendering Process.*

```
Create Particles Settings
For each particle
  Obtain particle's audio stream
  For each time instant during particle's lifetime
    Get audio sample value from particle's audio stream
    Calculate particle's position
    For each virtual microphone
      Calculate particle/microphone conditions
      Calculate propagation delay
      Calculate the final sample value
      Mix the value on the final microphone buffer
    End
  End
End
Apply Audio Post-processing
```

3.1. Particle's Audio Stream

During rendering, the audio stream of each particle is obtained, i.e., the audio stream that each particle will reproduce. In some situations it may be as simple as randomly choosing one of the available audio streams, but in other situations it may represent additional signal processing. For instance, a particle may reproduce only a 1 kHz 1 octave band. As such, the signal processing that is required must be applied. Also, if the original audio streams have sample rates that are different from each other or different from the sample rate that the rendering system is using, additional sample rate conversion is required.

3.2. Particle-Microphone conditions

In each instant, the relation between the particle and the microphone will affect the way the particle sound is captured: their distance will be used to apply a distance attenuation; and angle between them must be taken into consideration for applying the polar response of the microphone.

In some situations, these angles between them may also be responsible for additional digital processing (e.g. a binaural virtual microphone that uses Head-Related Transfer Functions).

Due to propagation delay, what is being reproduced by a particle is not instantaneously captured by the microphone. As such, time related information must be considered - on simpler scenarios, a simple delay shift is enough, but if the particle is moving at a significant velocity, Doppler effects must be taken into consideration, which will require some sort of additional signal processing (e.g. interpolation, resampling, etc).

3.3. Audio Post-Processing

A particle system for audio applications requires an extensive dynamic range, because the system may range from a small number of particles, located at a great distance, reproducing audio streams of low amplitude; to a huge number of particles, located at a very small distance from the microphone, reproducing high level audio streams. As such, after mixing the sound of all particles over time in all microphones, it is important to apply some sort of peak normalization, before exporting the audio result to an audio file or for audio listening.

Also, some virtual microphones setups may require some additional post-processing (e.g. a MS microphone pair that needs to convert the buffer from Mid-Side to Left-Right).

3.4. Propagation Conditions

In most cases, the rendering engine will consider the usual settings regarding sound velocity (around 340 m/s) and open-field propagation attenuation (inverse-square law, with -6dB with distance doubling). Nevertheless, in some situations it may be important to change these values, either for creative effects or even to achieve some modeling goals. Both values can even be disabled. Disabling the sound speed will result in the sound of each particle arriving instantaneously to the virtual microphones, and removing any time related effect due to propagation. By disabling the propagation attenuation, the distance between the particle and the microphone will not impact the particle sound gain value.

3.5. Computational requirements

Due to several factors, it is highly recommended to use of double precision calculations (aka double), to prevent losing information caused by the high number of operations and their impact on dynamic range.

As can be easily seen, the rendering process will require significant computational power, especially in systems with a large number of particles and with substantial digital processing by particle (FIR filters, time/frequency/resampling processing, etc.).

As in computer graphics, current GPUs can also be used for sound rendering proposes, using their DSP-like architecture to decrease the required rendering time.

The order of the three main cycles of Algorithm 1 (particle/time/microphone) can be changed, to obtain a better code optimization or better user interaction. Depending on the implementation and the required use of memory (and their eventual disk access due to virtual memory constraints), the algorithm may benefit from some code optimization. Also, instead of forcing a full rendering, the system may allow the user to suspend the ren-

der process at some point, to allow him/her to listen to the results of the first seconds.

In some situations, especially when particles have a fixed position, the system may export an impulse response for each virtual microphone (by rendering a single impulse – dirac – as the input audio stream). These impulse responses can then be applied to different audio files or even used for real-time audio processing.

To lighten the burden of rendering all particles' conditions for each time instant, some time granularity may be applied, by calculating conditions in a periodic time (e.g. 50 ms) and maintaining such conditions within each time fragment or by using an interpolation method. Even with transition mechanisms (e.g. cross-fades), the computational power optimization may be relevant without affecting too much the final audio quality.

4. APPLICATIONS

The concept of particle systems can be used on several audio applications (some audio examples can be found in http://www.estg.ipleiria.pt/~nuno.fonseca/papers/dafx2013). A small overview of some of these applications is presented.

4.1. Sound Design

One of the most obvious applications is sound design. The same way that particle systems are widely used on computer graphics to recreate visual effects (rain, fire, smoke, explosions, etc.), the same applies to sound.

Having many particles or even many particle systems simultaneously can be used to create highly complex and rich audio environments (e.g. a war battle, a city, a stadium, a tropical forest, etc.). But particle systems can also be used to create simpler sounds, using different particles to add different sound layers or to create sound repetitions.

One of the advantages of this approach is the 3D sound features. Although the system could be rendered with a single (mono) microphone, sound can be easily obtained with almost any surround microphone arrays or setups.

4.2. 3D Mixing/Processing

Since particles are spread over a virtual 3D space, it is also evident their application in 3D sound. By scattering particles over some areas of the 3D space and rendering the final result using virtual microphones arrays, 3D mixing can be obtained.

Although particle systems are based on the idea that the user does not control individual particles, in some 3D sound applications it might be interesting for the user to place spot particles, and still use the rendering features of the system. This can be obtained by using several particle systems with only 1 particle within each system. With this approach, and narrow parameters (for instance, forcing the starting position at a point instead of a larger area), the software that was original conceived for particle systems, may be used for rendering single particles in space.

Position modifiers can also be used to add movement to sound, allowing the creation of special spatial effects.

By applying different virtual microphones setups, the same scene can be rendered in different output formats (e.g. mono, stereo, binaural, ambisonics, 5.1, 6.1, etc.).

4.3. Reverb

By considering that each particle can be "viewed" as the result of a sound reflection, it is possible to use a particle system as a reverb design tool. A particle located at a distance of 1 km, will not represent the reflection from a wall located at that distance, but represents a sound that had to travel 1 km (between reflections) until being capture by the microphone.

To implement this type of application, each particle will be static (no particle movements), the number of particles of the system must be very high (to prevent time "holes"), filter-based audio modifiers would enhance the output result (by simulating the different frequency response from different reflection materials), and particles should cover a very large area (or the "virtual" speed of sound should be decrease).

Although this type of application was a big computational cost, due to the huge number of used particles, the output of the system can be exported as an impulse response.

4.4. Granular Synthesis

It was in the context of granular synthesis that existing work regarding the use sound with particle systems, were used, since the relation is easily seen: one audio grain corresponds to a particle. In fact, the use of particle systems in computer graphics, for generating effects like rain or smoke, acts as a graphical granular synthesis, where many image grains create visual effects that are difficult to create with traditional surface-based modelling techniques.

In this type of scenario, each particle would be responsible to reproduce only a very small audio fragment (grain), using durations between 10 and 50 ms. By feeding the particle system with several audio files, each particle would use an audio modifier that would reproduce only a very short fragment of one of the available files. Also, many particles are required to cover the intended time duration without sound "holes" or using a sequential creation feature that creates particles at a fixed rate.

Once again, the system allows very interesting features regarding the use of granular synthesis with surround/multichannel applications.

4.5. Up-Mixing

Within the 3D sound applications, particle systems may be used as an up-mixing tool, using audio material that was recorded/created with a small amount of audio channels (mono, stereo) and allow that material to "shine" with a wider sound image.

By simply using filter-based audio modifiers and spreading the original sound on a 3D space and capturing the virtual sound, we can get up-mixing results.

Since each particle will be considered only as a very small component of the original sound, particle movement can also increase sense of spaciousness, especially if that movement is not uniform among particles.

If time-related effects (delay/reverb) are not desired, the system only has to make sure that every particle is at the same distance from the microphones (circle line), which will result in capturing the sound of each particle at the exact same time.

4.6. Impulse Response Up-Mixing

A particle system can also be used to increase the number of channels (up-mix) of a room impulse response (e.g. converting from mono to 5.1).

To achieve that, a small reverse-engineering process must be done: for each coefficient of the impulse response, one or more particles should be created at random locations, but at a distance that corresponds to that coefficient time instant delay, and with an overall gain that correspond to the coefficient amplitude value. With this approach, if a single impulse (dirac) audio stream is used in all particles, an omnidirectional virtual microphone will render the exact same impulse response. Now it is only a matter of replacing the omnidirectional microphone, by a multichannel microphone array, to obtain the impulse responses for the different channels.

5. CONCLUSIONS

This paper presented a conceptual model for the use of particle systems in audio applications, using a fully rendered system with virtual microphones.

By creating a full dynamic system, new possibilities appear, allowing total consistency between the obtained sounds and their space information, something that the use of simple random number generators are not able to get, especially with multichannel applications.

Although the concept of particle systems are seldom used in audio applications, they may have very high potential in several sound related areas, by mixing features related to 3D sound, sound design, reverberation, filtering and granular synthesis.

In the future, each one of the presented applications must be deeply studied, at an independently manner, and compared with current state-of-the-art methods within that application field.

6. REFERENCES

[1] J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes, "Computer graphics: principles and practice", 2nd ed., Addison-Wesley Longman Publishing, 1990.

[2] W.T.Reeves, "*Particle Systems—a Technique for Modeling a Class of Fuzzy Objects"*, ACM Transactions on Graphics (TOG), Volume 2 Issue 2, pp. 91-108, April 1983.

[3] D.Kim-Boyle, "*Sound Spatialization with Particle Systems*", in Proc. 8th International Conference on Digital Audio Effects (DAFX-05), Madrid 2005, pp. 65-68; Proceedings of the Journées d'informatique musicale Conference, Paris 2005, pp. 143-146.

[4] D.Kim-Boyle, "*Spectral Spatialization - An Overview*", in Proc. 2008 International Computer Music Conference, Belfast, 2008.

[5] P.R.Cook, "Modeling Bill's Gait: Analysis and Parametric Synthesis of Walking Sounds", in Proc. 22nd International Conference (Virtual, Synthetic, and Entertainment Audio), Audio Engineering Society, 2002.

[6] U. Zölzer, X. Amatriain, D. Arfib, J. Bonada, G. De Poli, P. Dutilleux, G. Evangelista, F. Keiler, A. Loscos, D. Rocchesso, M. Sandler, X. Serra, T. Todoroff, *DAFX:Digital Audio Effects*. John Wiley & Sons, pp. 174, May 2002.

[7] C. Verron,G. Drettakis, *Procedural Audio Modeling for Particle-Based Environmental Effects*, in *Proc. 133rd Audio Engineering Society Convention*, 2012.