Lightweight Scene-aware Rain Sound Simulation for Interactive Virtual Environments

Haonan Cheng*
Communication University of China
Tianjin University

Shiguang Liu†
Tianjin University

Jiawan Zhang‡
Tianjin University

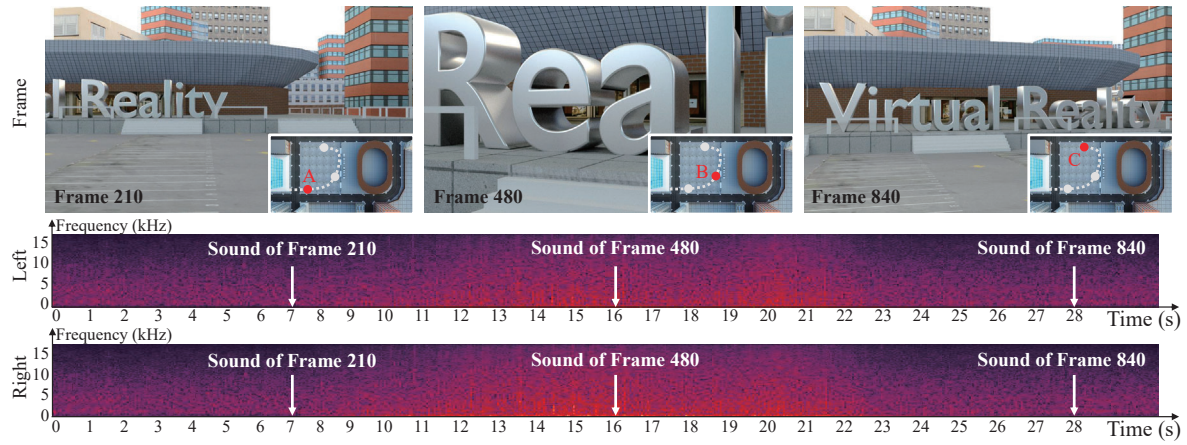

Figure 1: **Illustrations of the audiovisual simulation result of roaming in the square on a rainy day.** The top row presents three frames of the visual simulation results of the listener in the three different positions. The bottom right corner inset in each frame shows the corresponding airview map and the red dot indicates the listener's current position. As the user's distance from the metal "Virtual Reality" changes, our method is able to capture this dynamic change of metal sound effects as shown in the spectra (from top to bottom are left and right channels) on the bottom row. The white arrows mark the generated audio synchronized with the visual animation.

Abstract

We present a lightweight and efficient rain sound synthesis method for interactive virtual environments. Existing rain sound simulation methods require massive superposition of scene-specific precomputed rain sounds, which is excessive memory consumption for virtual reality systems (e.g. video games) with limited audio memory budgets. Facing this issue, we reduce the audio memory budgets by introducing a lightweight rain sound synthesis method which is only based on eight physically-inspired basic rain sounds. First, in order to generate sufficiently various rain sounds with limited sound data, we propose an exponential moving average based frequency domain additive (FDA) synthesis method to extend and modify the pre-computed basic rain sounds. Each rain sound is generated in the frequency domain before conversion back to the time domain, allowing us to extend the rain sound which is free of temporal distortions and discontinuities. Next, we introduce an efficient binaural rendering method to simulate the 3D perception that coheres with the visual scene based on a set of Near-Field Transfer Functions (NFTF). Various results demonstrate that the proposed method drastically decreases the memory cost (77 times compressed) and overcomes the limitations of existing methods in terms of interaction.

Index Terms: Human-centered computing—Human computer interaction—Interaction techniques—Auditory feedback; Applied computing—Arts and humanities—Sound and music computing

*e-mail: haonancheng@cuc.edu.cn
†e-mail: lsg@tju.edu.cn
‡Corresponding author. e-mail: jwzhang@tju.edu.cn

1 Introduction

Rain sound, as a common environmental sound, is widely used in video games and virtual reality (VR) applications. Most commonly, the rain sound in these virtual scenes is homogeneous and stationary, which can only provide constant information over a period of time. However, raindrops hit different surfaces such as umbrellas or trees can make different sound effects. If we simulate all sounds generated by raindrops as a single type of rain sound, the spatial information of virtual environment will be ignored. In this context, scene-aware rain sound, which refers to synthetic rain sound that interacts with scene dynamics, can convey more messages about the environment to listeners and enhance the realism of virtual scenes. For instance, when the listener moves towards the metal advertising words (see Figure 1), more and more clear metal sound effects can enhance the listener's sense of distance. Moreover, when the listener's view angle is changed and the metal advertising words are out of sight, the remaining metal sound effects from behind may remind the listeners of the nearby metal advertising words.

To achieve this goal, several researchers were attracted to explore the scene-aware rain sound synthesis methods [22, 42, 46]. Zita [46] proposed a physically-based method by analyzing the relationship between raindrop kinetic energy and velocity. However, the prohibitive simulation cost is not feasible for large virtual environments. In order to improve the efficiency of rain sound simulation, Verron and Drettakis [42] designed a signal-based rain sound model with five physically-inspired "sound atoms". This method is highly efficient, however, the synthesis parameters for sound modelling cannot

be fully automatically extracted from recorded sounds. Lately, an efficient scheme [22] was proposed to synthesize realistic surface material-aware rain sound. A basic rain sound bank was created for various surfaces during pre-computation. However, the basic rain sounds must reside in memory because the latency of streaming from disk is too high. As such, the basic rain sound bank is a memory-inefficient representation (about 100 MB) which is far more than the sum of all audio memory budgets in video games [23]. Moreover, the method [22] is ill-suited for interactive virtual environments due to that a smoothing algorithm is needed to clear up sound artifacts in the post-processing.

In this context, there still exist several challenges that are under-explored. Firstly, how to generate sufficiently various rain sounds with limited sound data is the key issue for the memory challenge. Different numbers of raindrops hitting different surfaces can emit various rain sounds, while the audio memory budget of video games and VR applications is limited. For example, there is only 25 MB of audio memory budget for all audio and 2 MB of audio memory budget for impact sounds in the open-world video game Crackdown II [23]. Secondly, how to automatically couple and render a large number of dynamic sound sources in real time is another burning issue. Video games and VR applications require real-time sound feedback, while the rainfall scenario usually contains a large number of dynamic sound sources which will bring in heavy computation. Moreover, a real-time feedback system not only requires high computation efficiency, but also cannot be optimized through manual adjustment or post-processing.

Focusing on these issues, in this paper, we propose a lightweight scene-aware rain sound simulation method that can efficiently reduce memory consumption and synthesize interactive rain sound in real time without any manual adjustment or post-processing. An example generated by our sound synthesis system is illustrated in Figure 1 (see more in Section 4). Firstly, to solve the memory challenge, we propose an exponential moving average based frequency domain additive (FDA) synthesis method based on eight basic rain sounds which satisfied the requirements of low memory budget. The eight basic rain sounds are short rain sound clips corresponding to the sounds of raindrops falling on different surfaces. Specifically, the exponential moving average based FDA rain sound synthesis algorithm contains two parts: the FDA synthesis method is designed to realize the variety of rain sounds and the exponential moving average based concatenation method is proposed to extend the short rain sounds with no sense of repetition. In doing so, we drastically decrease the memory budget (1.3 MB in total) compared to the previous method [22] (about 100 MB). Subsequently, an efficient binaural rendering method is proposed to ensure the real-time audio feedback and simulate the 3D perception that coheres with the visual scene. Specifically, a set of low order Near-Field Transfer Functions (NFTF) with simple structure are utilized to approximate the binaural effects. In summary, the main contributions of our work are listed as follows:

- We propose a lightweight scene-aware rain sound simulation method for interactive virtual environments. Our method can drastically decrease (77 $\times$) the audio memory budgets compared with the state-of-the-art sound synthesis solutions.
- A novel FDA rain sound synthesis algorithm based on exponential moving average is proposed to generate and extend short sound clips. With this approach, long segment of sufficiently various rain sounds can be generated with no sense of repetition.
- An efficient NFTF-based binaural rendering method is designed for real-time audio feedback and fine scale audio-visual consistency. The proposed scheme allows the simulation of a large number of sound sources with a low computational cost.

2 Related Work

In order to improve the sense of realism and immersion, in recent years, more and more attention has been drawn to sound synthesis in virtual environments. In this section, we briefly cover the background on sound simulation, propagation and binaural rendering related to our work.

2.1 Sound Simulation

Sound simulation has a long history in categories of computer animation [40], sound and music computing [7, 13], and interactive virtual environments [11, 12]. A general methodology for producing sound effects for animations was addressed in [40]. O'Brien *et al.* [29] targeted on the sound generation corresponding to the motions of solid objects. Later, several methods [5, 20, 24, 27, 36, 37, 45] focused on the nonlinear sound generated by tree movement, thin shells, liquid, elastic rods, paper, etc. Zheng and James [45] proposed a practical method to synthesize synchronized harmonic bubble-based sounds for 3D fluid animations. Due to the complexity of the fluid motion and bubble calculation, this method is costly at runtime. Concurrent with [45], Moss *et al.* [27] addressed the liquid sound synthesis problem with different types of fluid simulators and achieved real-time effects on shallow water sound simulation. Chadwick *et al.* [5] explored the nonlinear sound model for thin-shell vibrations. Although some simplified schemes are adopted in the method, this method still can't achieve real-time feedback. Langlois *et al.* [20] further explored complex acoustic bubbles and proposed a liquid sound synthesis method directly from two-phase incompressible simulations. This method further enhances the realism of the liquid sound, but it also requires more computational cost. Schweickart *et al.* [37] proposed the sound simulation method of rod-like structures. However, the internal mesh representation increased memory and computation time requirements. For the non-interactive scenes such as animations, several methods had been proposed to synthesize more and more realistic sound. Although sounds generated from complex nonlinear phenomena can be simulated, the heavy computational load prevents the use of the method in interactive settings.

Compared to sound synthesis methods for animation, methods for virtual environments and video games put forward higher requirements in terms of computing time complexity and space complexity. James *et al.* [19] proposed a timbre trees based method to produce convincing sound effects in an integrated way. Later, Doel *et al.* [12] simulated the sound of objects colliding, rolling and sliding through modal resonance models. As early explorations, these methods provide us a pipeline for parameterizing synthetic synchronized sounds. Later, a lot of research emerged to explore the real-time synthesis of different sounds. Dobashi *et al.* [9] proposed an efficient method for simulating aerodynamic sound created by the motion of fluids. They [10] further explored the vortex sounds corresponding to the motion of fluids and synthesized the sound at interactive rates. Carlo and Davide [4] proposed a cartoonication particle-based liquid sound rendering method. This method focuses more on the prototypical characteristics of the events rather than on the typical acoustic realism of the audio sample. To balance realism and computational cost, granular-based sound synthesis methods [8, 39] are widely used in virtual reality scenarios with real-time feedback. Various sound synthesis methods for several different sounds such as footstep sound [28], ocean sound [44], liquid sound [6] are explored for the virtual reality system.

Among different kinds of sound, sound effect for the rainfall scenario is still less explored, mainly due to that the rain sound is traditionally treated as a homogeneous and stationary background sound effect. To explore the scene-aware rain sound, Zita [46] proposed a physics-based model to calculate the rain drop sound from rigid surface and soft surface. However, since rainfall scenario is usually large scale with a large number of raindrops, this method can not achieve real-time playback due to the expensive physical

simulation. To achieve efficient computing performance, Verron and Drettakis [42] proposed a physically-inspired, signal-based rain sound model, yet the synthesis parameters for sound modelling cannot be fully automatically extracted from recorded sounds and the synthesized rain sound is less synchronized with the high randomness rain scene. Recently, Liu *et al.* [22] designed a physically rain sound simulation method based on acoustic characteristics of rain sound [15, 17, 25, 30–32]. To enable efficient sound simulation, a basic rain sound bank is precomputed using clustered sound models. However, the sound bank is about 100 MB to ensure the variety of sounds, which is memory-inefficient and ill-suited for audio memory budgets limited virtual reality systems. Besides, this method needs the post-processing to smooth the discontinuities during stitching different rain sound sources. Overall, current rain sound synthesis methods are ill-suited to generate high quality scene-aware rain sound in interactive virtual environments. To this end, it is necessary to develop a lightweight, interactive way for rain sound simulation.

2.2 Sound Propagation and Binaural Rendering

Sound propagation and binaural rendering are two important stages for delivering the spatial information to a binaural listener in a virtual environment. Sound propagation refers to the simulation of reflection, transmission, and diffraction paths from a sounding object to a listener's ear. Several geometric acoustic techniques [1,3,21,35], wave-based methods [26,34,41] have been proposed for sound propagation modelling. Wave-based techniques can simulate all acoustic effects accurately, yet usually is mathematically and computationally expensive. Geometric techniques can provide an efficient solution which enables real-time acoustic modelling. Due to the large-scale and highly dynamic characteristics of interactive virtual rain scene, we take advantage of a recent geometric technique [3] to simulate the sound propagation.

Binaural rendering refers to the 3D sound rendering which aims to simulate the sensation of binaural hearing. In virtual audio display, the head related transfer functions (HRTFs) are usually utilized to describe the sound perception. At present, several research works [33, 38, 43] have explored the HRTFs for the virtual audio display. Due to real-time requirements of interactive virtual environments, the brute-force computational approaches [16] are not appropriate to the application. Recently, an effective alternative [38] was proposed through a set of low-order parametric filters. This method can simulate the near-field acoustic effects at low computational costs, which is applicable to rain scenes with multiple sound sources. Different from Spagnol *et al.* [38], we take advantage of the low-order parametric filters and proposed a modified NFTF-based binaural rendering method which is tailored to large-scale and multiple sound sources rainfall scenarios.

3 Lightweight Scene-aware Rain Sound Simulation

Scene-aware rain sound can be synthesized through pre-computing different categories of basic rain sounds. However, the basic rain sound bank in previous approach is too large (about 100MB) for video games and interactive virtual reality applications. As such, our goal is to develop a lightweight method for scene-aware rain sound simulation. In this section, we start from the lightweight rain sound bank construction based on the similarity analysis of rain sounds generated through physical model. Then we describe the FDA synthesis method and exponential moving average based concatenation method which help us achieve sufficient variation with no sense of repetition based on the lightweight rain sound bank. Finally, we introduce the NFTF-based binaural rendering technique which is designed and utilized for 3D rain sound generation.

3.1 System Overview

As illustrated in Figure 2, our system works in four stages. Firstly, we construct a lightweight rain sound bank and define the pre-computed

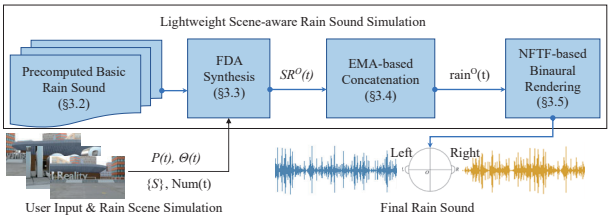

Figure 2: The architecture for our lightweight scene-aware rain sound synthesis technique. In each time step t, the user inputs the spatial position $P(t)$ and spatial angle $\Theta(t)$ by the input device (mouse, keyboard, etc.) and the visual simulation module provides sound source positions $\{S\}$ and raindrop number $Num(t)$ to the sound simulation module. In the sound simulation module, there are four main components in the pipeline: precomputed short rain sound as basic rain sound, FDA synthesis for generating target rain sound, exponential moving average based concatenation for extending short rain sound clips and NFTF-based binaural rendering for 3D sound generation. The output of the sound simulation module is the 3D rain sound that coheres with the visual scene.

rain sounds as basic rain sounds. The pre-computed basic rain sounds are very short. As such, we can reduce the audio memory budget greatly. In the FDA synthesis stage, we reconstruct the target rain sound which is corresponding to the current specific rain scene based on the pre-computed basic rain sound. Specifically, the modification and superposition of basic rain sounds are guided by the input interaction parameters in the frequency domain. The input interactive parameters consist of two parts: spatial position $P(t)$ and spatial angle $\Theta(t)$ from the user input, sound source position S and raindrop number $Num(t)$ from the scene simulation. Then, to extend the short target rain sound, an exponential moving average based concatenation method is designed which can generate endless sound with no sense of repetition. Next, to achieve the 3D perception that coheres with the visual scene efficiently, we propose a NFTF-based binaural rendering to approximate the distance rendering and perception of virtual rain sound sources. Each major technical component of our system will be described in detail in the following sections.

3.2 Lightweight Rain Sound Bank Construction

In this section, our goal is to construct a sufficiently small rain sound bank which still can contain the sound characteristics of different surfaces (water, leaves, etc.) and rainfall intensities (light rain, heavy rain, etc.). Contrary to synthesize various types of rain sounds directly based on physical model [22], our core idea is to explore the minimal set of rain sounds, from which most of various types of rain sounds can be efficiently constructed. To this end, we analyze the scaling relationships among different rain sounds based on the audio feature similarity comparison.

Specifically, we synthesize rain sounds for both liquid surface and solid surface of different rainfall intensities based on the physical model [22]. The rainfall intensities are divided into ten intervals according to the raindrop numbers commonly used in rainfall simulation. The total raindrop number range is $5000 \sim 10000$ and the length of each interval is 500. Then, to explore scaling relationships among different rain sounds, we further calculate the linear superposition results of rain sounds in different intervals.

To analyze the similarity of linear superposition results and physical model outputs, we extract four audio features of these rain sounds, namely spectral centroid, spectral entropy, spectral flux and spectral roll off [1]. The feature curves of rain sounds of different rainfall intensities generated by physical model and linear superposition

[1] Spectral Centroid is the average frequency of signal weighted by magni-

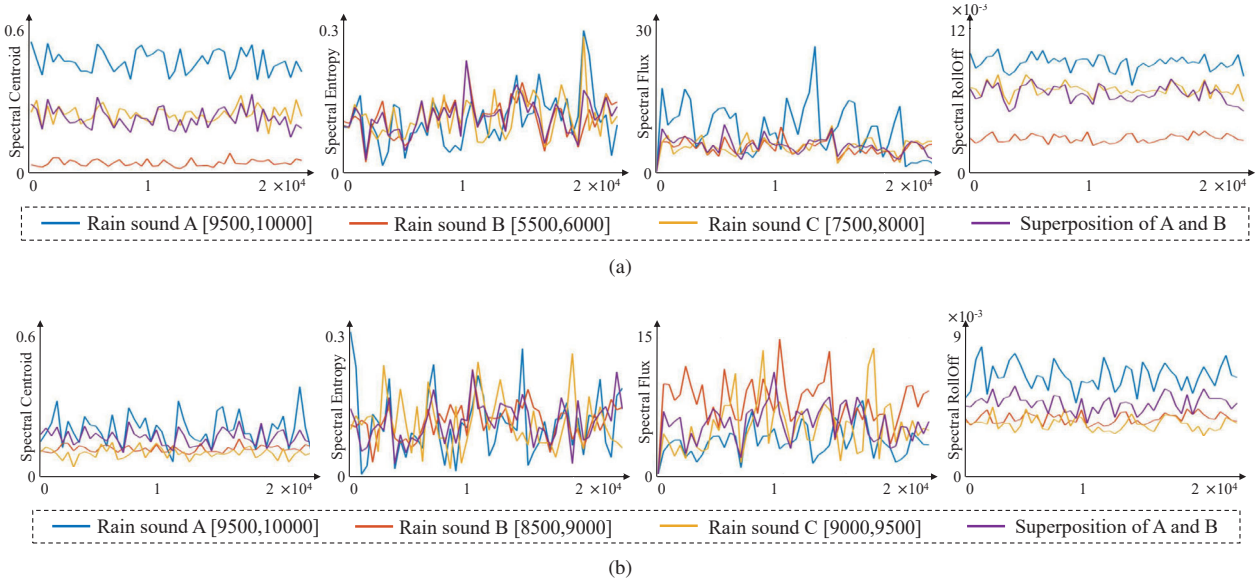

Figure 3: An illustration of audio features of sound generated by different methods. There are four feature curves in each sub-figure, three of which correspond to sounds synthesized by physical methods and one to sounds generated by linear superposition. The rain sound A[9500,10000] indicates that the number of raindrops interval is 9500 to 10000.

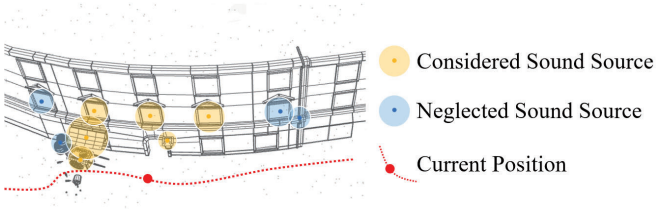

Figure 4: An illustration of the considered and neglected sound sources in a virtual scene. The red dotted line represents the listener's movement track. When the listener is at the red dot, the surrounding yellow sound sources will be calculated.

are illustrated in Figure 3. It can be observed that through the linear superposition of sound, we can get the similar sound generated by the physical model. Especially in the larger scale (Figure 3 (a)), we can clearly observe that the spectral centroid curve and spectral roll off curve of superposition result is very similar to the curves of the physical model result. This suggests that the energy distribution of these two sounds is very similar. When in the smaller scale, because the four kinds of sounds are similar, the similarity characteristics are not so obvious.

Therefore, we can only synthesize and store the rain sound clips at both ends for the two categories of sounds: impact sound for solid surface, the bubble sound for liquid surface. Based on the aforementioned observation and analysis, we finally store four basic rain sounds, namely the light impact sound $IL(t)$ in [5000,5500] and heavy impact sound $IH(t)$ in [9500,10000], and the light bubble sound $BL(t)$ in [5000,5500] and heavy bubble sound $BH(t)$ in [9500,10000]. Please see Appendix A for the synthetic process of the basic rain sound. In order to further distinguish the sound of raindrops falling on solid surfaces of different materials, we store four typical materials sound recordings from BR-sound bank [22]: glass sound $MG(t)$, metal sound $MM(t)$, wood sound $MW(t)$ and cloth sound $MC(t)$. The rain sounds with material sound textures are synthesized through a signal reconstruction model (see sec 4.3 of [22]). Therefore, there are eight rain sound clips stored in total in pre-processing. Each sound clip is 0.5 seconds long and about 170 KB. Thus the total audio memory budget is about 1.3 MB, which has greatly reduced the audio memory budgets.

tude, Spectral Entropy calculate the regularity of power spectrum of signal, Spectral Flux differentiate between normalized spectral magnitudes, and Spectral RollOff measures the spectral concentration less than threshold.

3.3 FDA Synthesis

The pre-computed rain sounds are homogeneous and stationary, so they cannot provide consistency with the current visual scene and user position. To synthesize the rain sound that coheres with the expected human perception, we present a FDA synthesis method which can generate different rain sound sources based on the lightweight sound bank. Although time-domain synthesis is faster, granular synthesis methods in the time domain are required to address the unpredictability of the phase at the beginning and at the end of the grains. In contrast, the advantage of synthesis in the frequency-domain is that the phase shift or magnitude of a signal at each frequency can be handled more intuitively. The additive synthesis in frequency-domain can reconstruct phase and provide continuity between consecutive grains, which may contain significantly different sonic data.

Specifically, in each time step t, we can get the parameters of listener spatial position $P(t) = (p_x(t), p_y(t), p_z(t))$ and spatial angle $\Theta(t) = (\theta_x(t), \theta_y(t), \theta_z(t))$ from the input device. In order to determine the location and strength of the rain sound source, the number of raindrops Num(t) in the simulated virtual environment and the positions of object's surfaces $\{S\}$ are also as the input to our sound synthesis algorithm. Then, we calculate the weight value of linear superposition by distance and number of raindrops. Each object surface is regarded as a sound source. Since human auditory distance is limited, we do not consider every object in the scene as a sound source and neglect the rain sound sources outside a fixed distance (as illustrated in Figure 4). The considered region is a hemisphere with the radius, where the radius can be set by users. In the experiment, to ensure the fairness of the comparison, we set the radius to 10

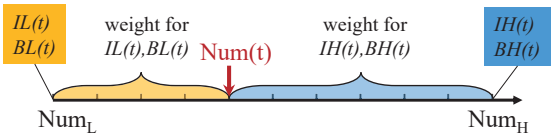

Figure 5: An illustration of the superposition ratio in distance interval. The proportion of blue area represents the weight of $IH(t), BH(t)$, and the proportion of yellow area represents the weight of $IL(t), BL(t)$.

which is the same as previous work. This allows us to efficiently calculate the rain sound of virtual scenes with different sizes. For a surface S^O of object O, we calculate the Euclidean distance $\mathrm{Dis}^{\mathrm{O}}(\mathrm{t})$ between the coordinates of surface (S_x^O, S_y^O, S_z^O) and the listener.

With the $\mathrm{Num}(\mathrm{t})$ and $\mathrm{Dis}^{\mathrm{O}}(\mathrm{t})$, we can determine the superposition ratio of the impact sounds and bubble sounds $\{IL(t), IH(t), BL(t), BH(t)\}$ in lightweight sound bank. As shown in Figure 5, the superposition follows a linear interpolation between $\mathrm{Num_L}$ and $\mathrm{Num_H}$ and the amplitude of rain sound is in inverse proportion with the distance $\mathrm{Dis}^{\mathrm{O}}(\mathrm{t})$. Moreover, the amplitude range of the material sound textures should be adjusted into a balanced scale with the impact sound and bubble sound. Take the near-field impact sound as an example, the amplitude of material sound texture is firstly scaled as follows:

$$M_{IN}^O(t) = \frac{\overline{AIN(t)}}{\overline{AM^O(t)}} \cdot M^O(t) \tag{1}$$

where $\overline{AIN(t)}$ indicates the average amplitude of the near-field impact sound $IN(t)$ and $\overline{AM^O(t)}$ is the average amplitude of material sound textures $M^O(t) = \{MG(t), MM(t), MW(t), MC(t)\}$. Finally, for an object surface S^O, the expression of the superposition rain sound $SR^O(t)$ is formulated as:

$$\begin{aligned} SR^O(t) = {} & [IL(t) + w \cdot BL(t) + M_{IL}^O(t)] \cdot \frac{\mathrm{Num_H} - \mathrm{Num(t)}}{(\mathrm{Num_H} - \mathrm{Num_L}) \cdot \mathrm{Dis}^{\mathrm{O}}(\mathrm{t})} \\ & + [IH(t) + w \cdot BH(t) + M_{IH}^O(t)] \cdot \frac{\mathrm{Num(t)} - \mathrm{Num_L}}{(\mathrm{Num_H} - \mathrm{Num_L}) \cdot \mathrm{Dis}^{\mathrm{O}}(\mathrm{t})} \end{aligned} \tag{2}$$

where $\mathrm{Num_H} - \mathrm{Num_L}$ is the raindrop number range of the both ends, $M_{IH}^O(t)$ refers to the scaled material sound textures for impact sound of heavy rain. For both solid and liquid surfaces, the sounds are synthesized based on the same basic rain sounds in varying proportions. When the surface is solid, the value of w is 0, otherwise, equals to 1. Furthermore, according to the efficiency shown by the previous frequency-domain approaches [2, 14], the above additive synthesis is also carried out in frequency-domain.

3.4 Exponential Moving Average based Concatenation

It is generally known that the rain sound is a kind of continuous and varying sound. Therefore, playing a looping short rain sound will damage the sense of reality and immersion. To solve this issue, we design an exponential moving average based connection method to extend and modify the generated short rain sound $SR^O(t)$. The main idea of the connection method is to smooth out the transitions between sound samples which acts as a low-pass filter. Therefore, we can compose the rain sound by simple and small acoustic grains in a continuous way. In this section, we first introduce the granular synthesis at spectral level, then we describe how we avoid discontinuities at the junction points.

Granular synthesis is a kind of overlap-add method which describes complex sounds as the composition of short and simple acoustic grains. Due to the high efficiency of this kind of algorithms, granular synthesis methods are proper for virtual environments to generate dynamic variations of the timbre endlessly. However, traditional time-domain granular synthesis methods usually connect sound samples with non-matching phases, which can often lead to audible clicks owing to the amplitude discontinuities.

Algorithm 1 Interactive rain sound simulation

Input: pre-computed basic rain sound: $\{IL(t), IH(t), BL(t), BH(t), MG(t), MM(t), MW(t), MC(t)\}$, position coordinates $(p_x(t), p_y(t), p_z(t))$, angle coordinates $(\theta_x(t), \theta_y(t), \theta_z(t))$, rain sound number $\mathrm{Num(t)}$, object surface S^O coordinates $(S_x^O, S_y^O, S_z^O), O \in \{\text{near-field and far-field region}\}$,
Output: 3D rain sound $\mathrm{rain_{3D}(t)}$

```
Determine the sound source field and update sound sources;
for all rain sound sources SR^O(t) ∈ sound source field do
    Calculate the Euclidean distance between listener and object surface S^O;
    Scale the material sound textures based on Equation (1);
    Synthesize rain sound source SR^O(t) based on FDA synthesis method (3.3);
    Normalize SR^O(t) to [-1,1];
    Generated rain^O(t) based on exponential moving average based connection method (3.4);
    Near-field sound effects simulation based on NFTFs and Far-field sound effects simulation based on BST algorithm (3.5);
end for
return rain_3D(t)
```

To avoid this issue, we note that exponential moving average filter in frequency domain can generate signal without jumping points. Specifically, the core idea of the designed exponential moving average based connection method is: the most recent sound samples are given more weight. First, we randomly select N consecutive samples from rain sound sources $SR^O(t)$. Let the selected samples $SP(t)$ as grains. Then, the selected samples $SP(t)$ and previous rain sound $\mathrm{rain^O(t-1)}$ of object O are converted into frequency signal by fast Fourier transform (FFT). To generate current rain sound $\mathrm{rain^O(t)}$ of object O, we perform the exponential moving average filter as follows:

$$\mathrm{rain^O(t)} = (1 - \omega_{EMA}) \cdot SP(t) + \omega_{EMA} \cdot \mathrm{rain^O(t-1)} \tag{3}$$

where ω_{EMA} is the weight coefficient of exponential moving average filter. Finally, the current rain sound $\mathrm{rain^O(t)}$ is converted into time domain through inverse fast Fourier transform (IFFT). When $t = 0$, the current rain sound $\mathrm{rain^O(t)}$ is the current rain sound sources $SR^O(t)$.

In summary, the designed exponential moving average based connection method (see Figure 6) is performed by the following steps:

- Randomly select N consecutive samples $SP(t)$ from rain sound sources $SR^O(t)$ as grains;
- Perform FFT of $SP(t)$ and previous rain sound $SR^O(t-1)$;
- Perform exponential moving average filter and generate the current rain sound $\mathrm{rain^O(t)}$;
- Take IFFT of the current rain sound $\mathrm{rain^O(t)}$.

We set $N = 9600$ and $\omega_{EMA} = 0.1$ in the experiment. By cycling the above process 5 times (the sampling frequency of synthetic sound is 48000 Hz), we can generate the varying rain sound $\mathrm{rain^O(t)}$. With

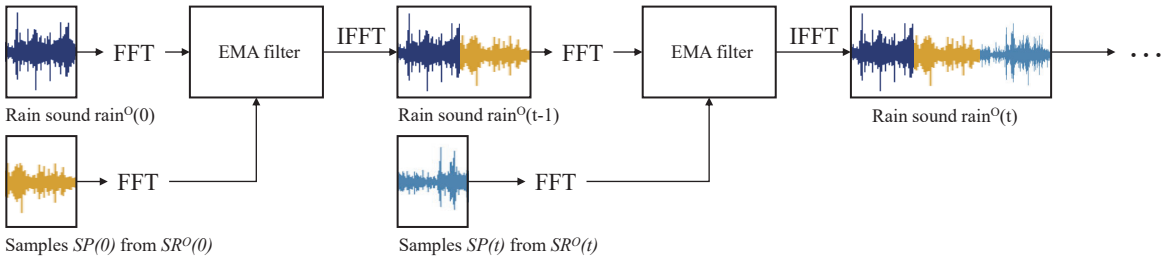

Figure 6: An illustration of the exponential moving average based connection pipeline.

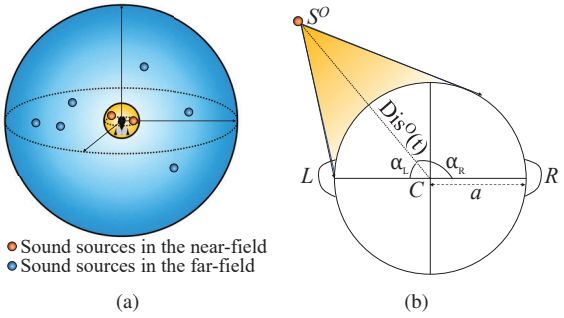

Figure 7: Illustrations of the virtual locations of sound sources (as dots) and head geometry. (a) shows the near-field (yellow) and far-field (blue) region. (b) illustrates the head geometry of a listener.

the exponential moving average based connection method, we can not only extend the rain sound endlessly, but also avoid the post-processing smoothing algorithm in previous work [22]. That makes it is well-suited for the real-time feedback interactive system.

3.5 NFTF-based Binaural Rendering

Since we aim to synthesize realistic scene-aware rain sound, in this case, the rain sound is no longer a background environmental sound. For example, the rain on the overhead canopy, the rain on right side metal show the spatial properties of sound sources (the sound effects are shown in the accompanying video). However, the rainfall scene is usually large scale with multi sound sources. As such, the high aural complexity of rainfall scene results in computational challenges for audio rendering.

To improve computing efficiency, we design a separate sound effects solution mechanise for near-field and far-field sound effect simulation (see Figure 7 (a)). For far-field sound propagation, we follow the bidirectional sound transport (BST) algorithm [3] at each time step. The BST method can handle multiple sources in large virtual environments at an interactive rate. For near-field binaural sound effect, we no longer simulate it in a brute-force computational way (finite-difference and finite methods). Instead, we adjust and integrate a set of NFTF filters to adapt them for our 3D rain scene.

Now we describe how we simulate the near-field sound effects through the NFTF-based binaural rendering. Due to the fact that the near-field HRTF databases are difficult to measure and construct, we need to find an efficient way to the realization of near-field virtual auditory display. To ensure the real-time feedback, we utilize a set of NFTF filters with simple structure and low order to accurately approximate the near-field sound effects. The NFTF filters are built upon incidence angle α and distance of source $\mathrm{Dis}^{\mathrm{O}}$. Taking the left ear as an example, the incidence angle is the angle between rays connecting the center of the head to the sound source and to the ear (see Figure 7 (b)). The incidence angle of left ear α_L can be calculated by the vector as follows:

$$\alpha_L(t) = acos\left(\frac{\overrightarrow{CS^O}\cdot\overrightarrow{CL}}{\left\|\overrightarrow{CS^O}\right\|\cdot\left\|\overrightarrow{CL}\right\|}\right) \tag{4}$$

Then we normalize the distance of source to head radius as follows:

$$\mathrm{Dis}^{\mathrm{O}}_{\mathrm{L}}(\mathrm{t}) = \left\|\overrightarrow{CS^O}\right\|/a \tag{5}$$

where a is the head radius and we fix the head radius to the standard value $a = 8.75$ cm. The distance and angle between the listener and the sound source are changing over time.

The spatialized rain sound $\mathrm{rain}_{3\mathrm{D}}(\mathrm{t})$ for near-field and far-field is synthesized by filtering the rain sound source $rain^O(t)$ through a pair of NFTF filters as follows:

$$\begin{aligned}\mathrm{rain}_{3\mathrm{D}}(\mathrm{t}) = &\sum_{O\in near-field} NFTF(\alpha(t), \mathrm{Dis}^{\mathrm{O}}(\mathrm{t}))\cdot\mathrm{rain}^{\mathrm{O}}(\mathrm{t}) \\ &+ \sum_{O\in far-field} BST(\mathrm{Dis}^{\mathrm{O}}(\mathrm{t}))\cdot\mathrm{rain}^{\mathrm{O}}(\mathrm{t})\end{aligned} \tag{6}$$

where $NFTF(\cdot)$ are the NFTF filters and $BST(\cdot)$ represents the BST algorithm followed [3]. The $NFTF filters$ utilized in Equation (6) are listed in Appendix B.

In the experiment, we set the calculation range of near-field simulation to 30 cm. This is because when $\mathrm{Dis}^{\mathrm{O}}_{\mathrm{L}}(\mathrm{t})$ (or $\mathrm{Dis}^{\mathrm{O}}_{\mathrm{R}}(\mathrm{t})$) is greater than 3, there is no significant change in sound intensity. In Section 4, we test and show the change of filters at different distances. In summary, the whole process of the interactive scene-aware rain sound simulation is listed in Algorithm 1.

4 Results and Discussions

We tested our method on various rain scenes. We implemented our algorithm on Matlab and all the experiments were conducted with the same hardware: Intel Core i5-4590, NVIDIA GeForce GTX 1660 Ti GPU, and 8GB RAM.

4.1 Comparison with the State-of-the-art Methods

In order to validate our interactive sound synthesis method, we compared it with the state-of-the-art methods, including a physical based method [22] and a signal based synthesis [42]. The spectra of generated sounds are illustrated in Figure 8. We can observe that our method is able to capture the dynamic change of rain sound when the listener goes under the canopy or out of the forest, while the method in [42] generates unchanged sound. Although the results of [22] also changes with the scene, it is noteworthy that our approach has a much smaller memory budget (77 $\times$) compared with [22] under the

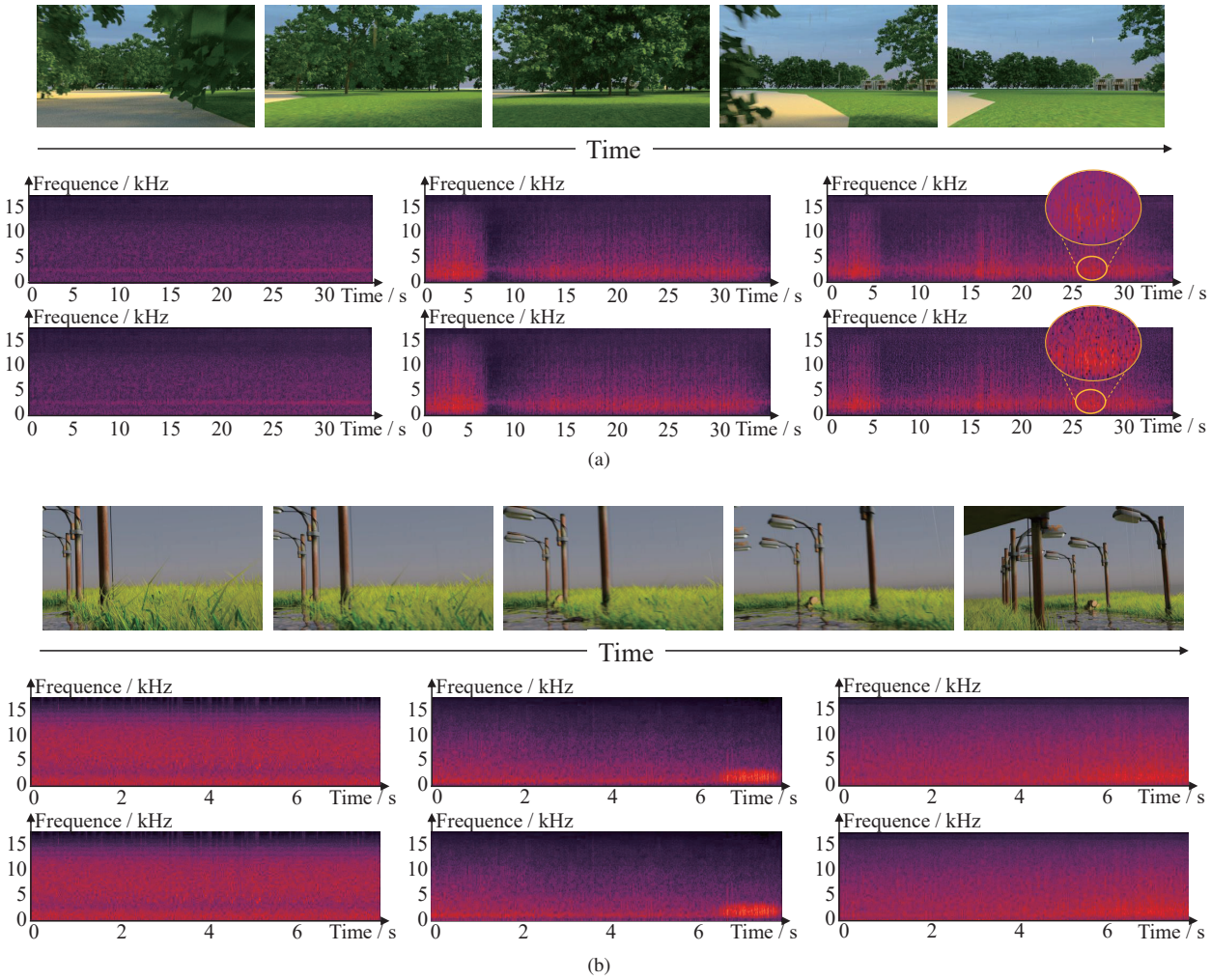

Figure 8: Comparison with the state-of-the-art methods. In each sub-figure, from top to bottom are: five frames of different positions, spectra of left channel and spectra of right channel. The spectra from left to right are the sound results of [42], sound results of [22] and our results, respectively.

condition of ensuring the high quality results and real-time feedback. Moreover, we simulated the binaural effects, which can be seen from Figure 8 that the spectra of left and right channels are different (see the circled parts in Figure 8), while [22] did not simulate the binaural effects. The sound difference between the left and right ear can be heard from the accompanying video.

4.2 Evaluation

In this section, we verify the efficiency of our algorithm and the selection of the parameters, which we will describe in detail in the following paragraphs.

4.2.1 Algorithm efficiency

We show the calculation time of two important stages and the total synthesis time for generating per second sound in Table 1. Moreover, to verify that our method saves memory without significantly increasing the computational overhead, we also compare it with the high memory consumption method [22]. Each column in Table 1 from left to right represents: the virtual scene, the size of the virtual scene, the length of the synthesized sound, the time for performing FDA and exponential moving average based concatenation method, the time spend performing NFTF filters and BST method, and the total sound synthesis time, the synthesis time of [22]. The virtual scenes of "Virtual Reality", "Countryside" and "Forest" correspond to Figure 1, Figure 8 (a) and Figure 8 (b), respectively. Form the table, we can observe that when the scale of the scene changes, our calculation cost does not vary greatly. This is due to that we set a fixed near-field and far-field range, and our calculation area will not expand with the increase of scene size. Besides, although the reduced memory causes the computation time of our method to be slightly higher than the previous work [22], our method still ensures real-time performance. In addition, we can see that both the FDA-EMA method for sound source expansion and the NFTF method for near-field sound effects simulation take a very short simulation time. Therefore, these two methods can be well integrated into other virtual reality algorithms.

4.2.2 Verification of the selection of near-field region

To determine the size of the near-field region, we constructed different NFTF filters with different normalized distance $Dis_L^Q(t)$. The results are illustrated in Figure 9, where we present the three dimensional diagram of the change of frequency intensity with the change of distance and angle. It is easy to observe that when the normalized distance $Dis_L^Q(t)$ is greater than 3, the intensity of frequency

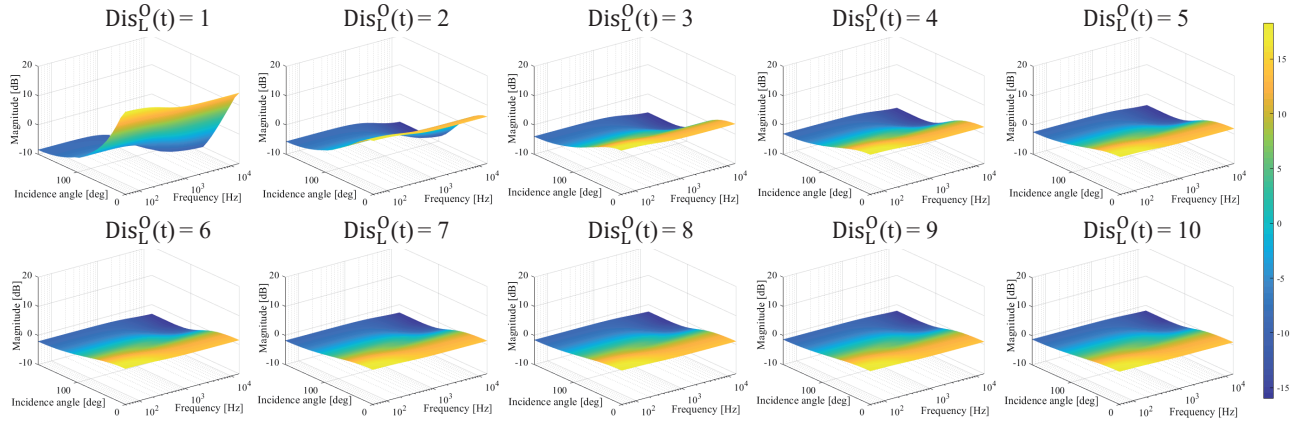

Figure 9: Spectra at different $\mathrm{Dis}_L^O(t)$ and different incidence angles. The $\mathrm{Dis}_L^O(t)$ is ranging from 1 to 10 and the angle is ranging from 0° to 180°.

Table 1: Timing for experiments with different scene sizes.

Scenarios	Domain size (m)	Length (s)	Time for FDA-EMA (ms)	Time for NFTF-BST (ms)	Our total synthesis time (s)	Synthesis time of [22] (s)
Virtual Reality	$30 \times 30 \times 11$	1.0	6.242	0.575	0.033	0.026
Countryside	$29 \times 29 \times 26$	1.0	5.507	0.552	0.026	0.025
Forest	$92 \times 60 \times 27$	1.0	5.092	0.594	0.031	0.026

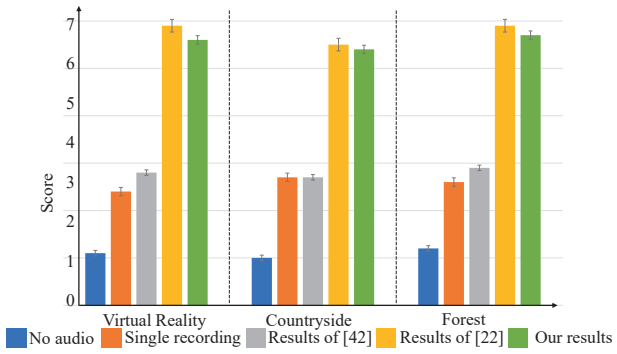

Figure 10: The bar chart showing mean value for the psychophysical experiment results comparison of scene with no audio, single recording, results of [42], results of [22] and our results. The error bar represents the data range within 1.5 $\times$ interquartile rang.

varies slightly. So we set $Dis_L^O(t)$ as 3 to avoid unnecessary extra calculations.

4.3 Psychophysical Experiment

To further evaluate the subjective expressiveness of our results, we conduct a double-blind multi-stimuli test follow the MUSHRA recommendation [18]. In each trial, the participant is presented with the reference, a recorded video for rainy day, and five virtual experiments in random order:

- Virtual environment without audio playback.
- Virtual environment with a single recording.
- Virtual environment with sound generated by approach [42].
- Virtual environment with sound generated by approach [22].
- Virtual environment with sound generated by our method.

17 subjects (10 male, 7 female, aged 23 to 50) participated in the experiment. All participants have either normal or corrected-to-normal vision and reported normal hearing ability, and were naive about purpose of the experiment.

During the experiment, the participants control themselves moving around the map by using the mouse and keyboard, and get audio feedback through headphones. Each of them is asked to answer the following three questions:

- Q1: Can you recognize what is happening outside your field of view?
- Q2: Do you think the sound matches the visual?
- Q3: How strong is the immersive feeling of the virtual environments?

For question 1 and question 2, we set 1 for "yes" and 0 for "no". For the third question, the score range is from 0 to 7.

In order to verify the immersion of our synthesized results, we first compare with the state-of-the-art online method [42] in interactive scenarios. Furthermore, the results of the third question are statistically analyzed with a paired T-test to verify whether there is significant difference between the score of our method and the score of the method [42]. We list the mean and standard deviation of each question scores, as well as the P value and T value of Q3 in Table 2, where the P value represents the probabilistic significance. From the table, we can observe that our scores are the highest in four situations, and the significance values also indicate that our result achieves significantly better sound effect than previous method [42] in the experiment. The dynamic change of rain sound has greatly enhanced the user's sense of immersion.

Subsequently, we compare the sound quality of the synthesized sounds by different methods based on the third question. The scores

Table 2: Subjective results of different scenes (four consecutive lines of the same background color correspond to a scene).

Scenarios	Mean (Q1)	Std (Q1)	Mean (Q2)	Std (Q2)	Mean (Q3)	Std (Q3)	T Value	P Value
No audio	0	0	-	-	1.00	0.612	-	-
Single recording	0.12	0.332	0.29	0.470	2.59	0.712	-	-
Sound results of [42]	0.18	0.393	0.35	0.493	2.71	0.772	-	-
Our sound results	0.29	0.470	0.59	0.507	4.94	0.748	27.253	*****
No audio	0	0	-	-	0.88	0.485	-	-
Single recording	0.06	0.243	0.24	0.437	2.24	0.437	-	-
Sound results of [42]	0.29	0.470	0.29	0.470	2.47	0.624	-	-
Our sound results	0.82	0.393	0.82	0.393	4.71	0.686	28.284	*****
No audio	0	0	-	-	0.94	0.556	-	-
Single recording	0.18	0.393	0.35	0.493	2.53	0.717	-	-
Sound results of [42]	0.41	0.507	0.47	0.514	2.35	0.702	-	-
Our sound results	0.82	0.393	0.82	0.393	4.47	0.514	35.827	*****

Significance codes (P): < 0.05: '***', < 0.01: '****', < 0.005: '*****'

of the experiment are illustrated in Figure 10. It can be observed that our approach can achieve similar performance to the offline approach [22] and is more efficient. Besides, large memory cost in our previous work [22] is caused by storing all pre-calculated rain sounds. Our improved lightweight rain sound synthesis benefits from similarity analysis of rain sounds and the exponential moving average based FDA method which generates and extends short sound clips. Therefore, our method can achieve comparable sound quality compared to the method based on larger budget.

5 Conclusions and Future Work

In this paper, we proposed a novel lightweight scene-aware rain sound synthesis method for interactive virtual environments. We took advantage of a series of simplification and approximation strategies in the process of synthesizing sound to ensure the real-time performance of the algorithm and reduce memory consumption. Specially, based on the exponential moving average based FDA synthesis method, we generated various endless rain sound with 8 short basic rain sounds. As a result, our approach enable us to reduce the memory budget 77 times. Besides, we simulated the near-field sound effects with a set of NFTF filters rather than complex physical simulation. Different scenarios were generated to evaluate the performance of our method. Subjective evaluation shows that our sound results can achieve equal variety of sound effects based on a much smaller memory budget and greatly enhance the realism and immersion of the virtual scene.

However, our method also has some limitations. First of all, the method of this paper still needs pre-computed sound samples and can not synthesize rain sound directly according to scene parameters. Constructing an efficient rain sound synthesis model will be more competitive in virtual reality applications and games. Secondly, we did not consider the influence of external forces on the rain sound, such as the effect of wind speed on the rain sound. Therefore, exploring how to integrate multi-environmental sound sources in virtual scenes is a research direction of our future work.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments. This work was supported by the Natural Science Foundation of China under grant No. 62201524, and the Fundamental Research Funds for the Central Universities under grant No. CUC22GZ002.

References

[1] C. R. Alla Chaitanya, J. M. Snyder, K. Godin, D. Nowrouzezahrai, and N. Raghuvanshi. Adaptive sampling for sound propagation. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1846–1854, 2019.

[2] N. Bonneel, G. Drettakis, N. Tsingos, I. Viaud-Delmon, and D. L. James. Fast modal sounds with scalable frequency-domain synthesis. *ACM Transactions on Graphics*, 27(3):24:1–24:9, 2008.

[3] C. Cao, Z. Ren, C. Schissler, D. Manocha, and K. Zhou. Interactive sound propagation with bidirectional path tracing. *ACM Transactions on Graphics*, 35(6):180:11–180:11, 2016.

[4] D. Carlo and R. Davide. Acoustic rendering of particle-based simulation of liquids in motion. *Journal on Multimodal User Interfaces*, 5:187–195, 2012.

[5] J. N. Chadwick, S. S. An, and D. James. Harmonic shells: a practical nonlinear sound model for near-rigid thin shells. *ACM Transactions on Graphics*, 28(5):119:1–119:10, 2009.

[6] H. Cheng and S. Liu. Haptic force guided sound synthesis in multisensory virtual reality (vr) simulation for rigid-fluid interaction. In *IEEE Virtual Reality*, pp. 111–119, 2019.

[7] P. Cook. Physically informed sonic modeling (phism): Synthesis of percussive sounds. *Computer Music Journal*, 21(3):38–49, 1997.

[8] S. Diemo. State of the art in sound texture synthesis. In *Conference on Digital Audio Effects*, pp. 1–11, 2011.

[9] Y. Dobashi, T. Yamamoto, and T. Nishita. Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. *ACM Transactions on Graphics*, 22(3):732–740, 2003.

[10] Y. Dobashi, T. Yamamoto, and T. Nishita. Synthesizing sound from turbulent field using sound textures for interactive fluid simulation. *Computer Graphics Forum*, 23(3):539–546, 2004.

[11] K. V. D. Doel. *Sound synthesis for virtual reality and computer games*. PhD thesis, The University of British Columbia, 1998.

[12] K. V. D. Doel, P. G. Kry, and D. K. Pai. Foleyautomatic: physically-based sound effects for interactive simulation and animation. In *ACM SIGGRAPH*, pp. 537–544, 2001.

[13] A. Farnell. *Designing sound*. The MIT Press, 2010.

[14] S. Fasciani. Spectral granular synthesis. In *International Computer Music Conference*, pp. 1–5, 2018.

[15] G. Franz. Splashes as sources of sound in liquids. *Journal of the Acoustical Society of America*, 31(8):1080–1096, 1959.

[16] P. Guillon, R. Zolfaghari, N. Epain, A. V. Schaik, C. T. Jin, C. Hetherington, J. Thorpe, and A. Tew. Creating the sydney york morphological and acoustic recordings of ears database. In *IEEE International Conference on Multimedia and Expo*, pp. 461–466, 2012.

[17] Y. Guo and J. Williams. A theoretical study on drop impact sound and rain noise. *Journal of Fluid Mechanics*, 227:345–355, 1991.

[18] ITU-R. Method for the subjective assessment of intermediate quality level of audio systems, 2015. Available: www.itu.int/rec/R-REC-BS.1534.

[19] K. H. James, G. Joe, W. L. Jong, G. Larry, T. Tapio, and M. Suneil. An integrated approach to motion and sound. *The Journal of Visualization and Computer Animation*, 6(2):109–123, 1995.

Table 3: Coefficients for NFTF filters [38].

α	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}
0°	12.97	-9.69	-1.14	0.219	-4.39	2.123	-0.55	-0.06	0.457	-0.67	0.174	-1.75	0.699
10°	13.19	234.2	18.48	-8.5	-4.31	-2.78	0.59	-0.17	0.455	0.142	-0.11	-0.01	-0.35
20°	12.13	-11.2	-1.25	0.346	-4.18	4.224	-1.01	-0.02	-0.87	3404	-1699	7354	-5350
30°	11.19	-9.03	-1.02	0.336	-4.01	3.039	-0.56	-0.32	0.465	-0.91	0.437	-2.18	1.188
40°	9.91	-7.87	-0.83	0.379	-3.87	-0.57	0.665	-1.13	0.494	-0.67	0.658	-1.2	0.256
50°	8.328	-7.42	-0.67	0.421	-4.1	-34.7	11.39	-8.3	0.549	-1.21	2.02	-1.59	0.816
60°	6.493	-7.31	-0.5	0.423	-3.87	3.271	-1.57	0.637	0.663	-1.76	6.815	-1.23	1.166
70°	4.455	-7.28	-0.32	0.382	-5.02	0.023	-0.87	0.325	0.691	4.655	0.614	-0.89	0.76
80°	2.274	-7.29	-0.11	0.314	-6.72	-8.96	0.37	-0.08	3.507	55.09	589.3	29.23	59.51
90°	0.018	-7.48	-0.13	0.24	-8.69	-58.4	5.446	-1.19	-27.4	10336	16818	1945	1707
100°	-2.24	-8.04	0.395	0.177	-11.2	11.47	-1.13	0.103	6.371	1.735	-9.39	-0.06	-1.12
110°	-4.43	-9.23	0.699	0.132	-12.1	8.716	-0.63	-0.12	7.032	40.88	-44.1	5.635	-6.18
120°	-6.49	-11.6	1.084	0.113	-11.1	21.8	-2.01	0.098	7.092	23.86	-23.6	3.308	-3.39
130°	-8.34	-17.4	1.757	0.142	-11.1	1.91	0.15	-0.4	7.463	102.8	-92.3	13.88	-12.7
140°	-9.93	-48.4	4.764	0.462	-9.72	-0.04	0.243	-0.41	7.453	-6.14	-1.81	-0.88	-0.19
150°	-11.3	9.149	-0.64	-0.14	-8.42	-0.66	0.147	-0.34	8.101	-18.1	10.54	-2.23	1.295
160°	-12.2	1.905	0.109	-0.08	-7.44	0.395	-0.18	-0.18	8.702	-9.05	0.532	-0.96	-0.02
170°	-12.8	-0.75	0.386	-0.06	-6.78	2.662	-0.67	0.05	8.925	-9.03	0.285	-0.9	-0.08
180°	-13	-1.32	0.45	-0.05	-6.58	3.387	-0.84	0.131	9.317	-6.89	-2.08	-0.57	-0.4

[20] T. Langlois, C. Zheng, and D. James. Toward animating water with complex acoustic bubbles. *ACM Transactions on Graphics*, 35(4):95:1–95:13, 2016.

[21] T. Lentz, D. Schröder, M. Vorländer, and I. Assenmacher. Virtual reality system with integrated sound field simulation and reproduction. *EURASIP Journal on Advances in Signal Processing*, 2007(1):1–19, 2007.

[22] S. Liu, H. Cheng, and Y. Tong. Physically-based statistical simulation of rain sound. *ACM Transactions on Graphics*, 38(4):123:1–123:14, 2019.

[23] D. Lloyd, N. Raghuvanshi, and N. Govindaraju. Sound synthesis for impact sounds in video games. In *Proceedings of Symposium on Interactive 3D Graphics and Games*, pp. 55–62, 2011.

[24] K. Matsuyama, S. Ota, M. Tamura, T. Fujimoto, K. Muraoka, and N. Chiba. Animating tree movement with sound effects generation. In *Eurographics*, 2003.

[25] H. Medwin, J. Nystuen, P. Jacobus, L. Ostwald, and D. Snyder. The anatomy of underwater rain noise. *Journal of the Acoustical Society of America*, 92(3):1613–1623, 1992.

[26] R. Mehra, N. Raghuvanshi, L. Antani, A. Chandak, S. Curtis, and D. Manocha. Wave-based sound propagation in large open scenes using an equivalent source formulation. *ACM Transactions on Graphics*, 32(2):19:1–19:13, 2013.

[27] W. Moss, H. Yeh, J. Hong, M. Lin, and D. Manocha. Sounding liquids: Automatic sound synthesis from fluid simulation. *ACM Transactions on Graphics*, 29(3):21:1–21:13, 2010.

[28] R. Nordahl, S. Serafin, and L. Turchet. Sound synthesis and evaluation of interactive footsteps for virtual reality applications. In *IEEE Virtual Reality*, pp. 147–153, 2010.

[29] J. F. O'Brien, P. Cook, and G. Essl. Synthesizing sounds from physically based motion. In *ACM SIGGRAPH*, pp. 529–536, 2001.

[30] A. Prosperetti, L. Crum, and H. Pumphrey. The underwater noise of rain. *Journal of Geophysical Research Oceans*, 94(C3):3255–3259, 1989.

[31] H. Pumphrey, L. Crum, and L. Bjørnø. Underwater sound produced by individual drop impacts and rainfall. *Journal of the Acoustical Society of America*, 85(4):1518–1526, 1989.

[32] H. Pumphrey and P. Elmore. The entrainment of bubbles by drop impacts. *Journal of Fluid Mechanics*, 220:539–567, 1990.

[33] K. Raghunath Rao and J. Ben-Arie. Optimal head related transfer functions for hearing and monaural localization in elevation: a signal processing design perspective. *IEEE Transactions on Biomedical Engineering*, 43(11):1093–1105, 1996.

[34] N. Raghuvanshi and J. Snyder. Parametric wave field coding for precomputed sound propagation. *ACM Transactions on Graphics*, 33(4):38:1–38:11, 2014.

[35] C. Schissler, R. Mehra, and D. Manocha. High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Transactions on Graphics*, 33(4):39:1–39:12, 2014.

[36] C. Schreck, D. Rohmer, D. L. James, S. Hahmann, and M. Cani. Real-time sound synthesis for paper material based on geometric analysis. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 211–220, 2016.

[37] E. Schweickart, D. James, and S. Marschner. Animating elastic rods with sound. *ACM Transactions on Graphics*, 36(4):115:1–115:10, 2017.

[38] S. Spagnol, E. Tavazzi, and F. Avanzini. Distance rendering and perception of nearby virtual sound sources with a near-field filter model. *Applied Acoustics*, 115(1):61–73, 2017.

[39] B. Stefano, D. M. Stefano, and D. R. The sound design toolkit. *SoftwareX*, 6:255–260, 2017.

[40] T. Tapio and K. H. James. Sound rendering. *ACM SIGGRAPH Computer Graphics*, 26(2):211–220, 1992.

[41] L. Thompson. A review of finite-element methods for time-harmonic acoustics. *Journal of the Acoustical Society of America*, 20(3):1315–1330, 2006.

[42] C. Verron and G. Drettakis. Procedural audio modeling for particle-based environmental effects. In *Proceedings of Audio Engineering Society Convention 133*, 2012.

[43] T. M. Voong and O. M. Auditory spatial perception using bone conduction headphones along with fitted head related transfer functions. In *IEEE Virtual Reality*, pp. 1211–1212, 2019.

[44] K. Wang, H. Cheng, and S. Liu. Efficient sound synthesis for natural scenes. In *IEEE Virtual Reality*, pp. 303–304, 2017.

[45] C. Zheng and D. James. Harmonic fluids. *ACM Transactions on Graphics*, 28(3):37:1–37:12, 2009.

[46] A. Zita. Computational real-time sound synthesis of rain. Master's thesis, Linköping University, 2003.

A Physically based Rain Sound Model

The physical modeling of rain sound is discussed in detail in [15,22], here we briefly introduce the calculation of Equation (1) and (2). The impact sound of a single raindrop is formulated as

$$p_I(r,t) = \frac{k^2 \rho c D_S(t)}{4\pi r} cos\theta e^{-ikr} \tag{7}$$

where ρ is the air density, c is the speed of sound in air, r is the distance from a listener to the sound source, t is time, k is the

wavenumber, θ is the polar angle and i is the imaginary unit, $D_S(t) = 0.05Q_S(t)$ is the dipole strength. $Q_S(t)$ is calculated through

$$Q_S(t) = A_I e^{-\beta_I t} sin(2\pi f_I t) \tag{8}$$

where the $A_I = \rho_0 c_0 V_I$ refers to the amplitude, V_I is the raindrop impact velocity, $\beta_I = f_I/0.5$ is the damping constant and f_I is the frequency (random assignment between 1kHz and 16kHz). V_I is calculated based on experimental approximation:

$$V_I = V_T \sqrt{1 - exp(\frac{-2gz}{V_T^2})} \tag{9}$$

$$V_T = \begin{cases} -17.8951 + 448.9498d + 16.3719d^2 - 45.9516d^3, \\ \qquad\qquad\qquad\qquad\qquad\qquad d \leq 1.4mm, \\ 24.1660 + 448.8336d - 75.6265d^2 + 4.2695d^3, \\ \qquad\qquad\qquad\qquad\qquad\qquad d > 1.4mm. \end{cases}$$

where d is the raindrop diameter which is randomly assigned in the interval [0.1mm, 5.8mm].

B NFTF Filters

In the NFTF model, Spagno *et al.* [38] approximate the model of near field acoustic effects with a low-order parametric filter as follows:

$$\begin{aligned} &NFTF_L(\alpha(t), Dis^O(t)) = \\ &H_{sh}(f, G_\infty(\alpha(t), Dis^O(t)), CF(\alpha(t), Dis^O(t))) \cdot G_0(\alpha(t), Dis^O(t)) \end{aligned} \tag{10}$$

where $H_{sh}(\cdot)$ is a first-order high-frequency shelving filter, $CF(\cdot)$ is the cutoff frequency, $G_\infty(\cdot)$ refers to asymptotic high-frequency gain and $G_0(\cdot)$ refers to the direct component gain. These are three polynomials based on the coefficients in Table 3 as follows:

$$G_0(\alpha(t), Dis^O(t)) = \frac{c_1 Dis^O(t) + c_2}{Dis^O(t)^2 + c_3 Dis^O(t) + c_4} \tag{11}$$

$$G_\infty(\alpha(t), Dis^O(t)) = \frac{c_5 Dis^O(t)^2 + c_6}{Dis^O(t)^2 + c_7 Dis^O(t) + c_8} \tag{12}$$

$$CF(\alpha(t), Dis^O(t)) = \frac{c_9 Dis^O(t)^2 + c_{10} Dis^O(t) + c_{11}}{Dis^O(t)^2 + c_{12} Dis^O(t) + c_{13}} \tag{13}$$

where the values of coefficients are the linear interpolation between adjacent angles according to the incidence angle $\alpha(t)$ in Table 3.