Real-Time Rendering of Decorative Sound Textures for Soundscapes

JINTA ZHENG, Oregon State University, USA
SHIH-HSUAN HUNG, Oregon State University, USA
KYLE HIEBEL, Oregon State University, USA
YUE ZHANG, Oregon State University, USA

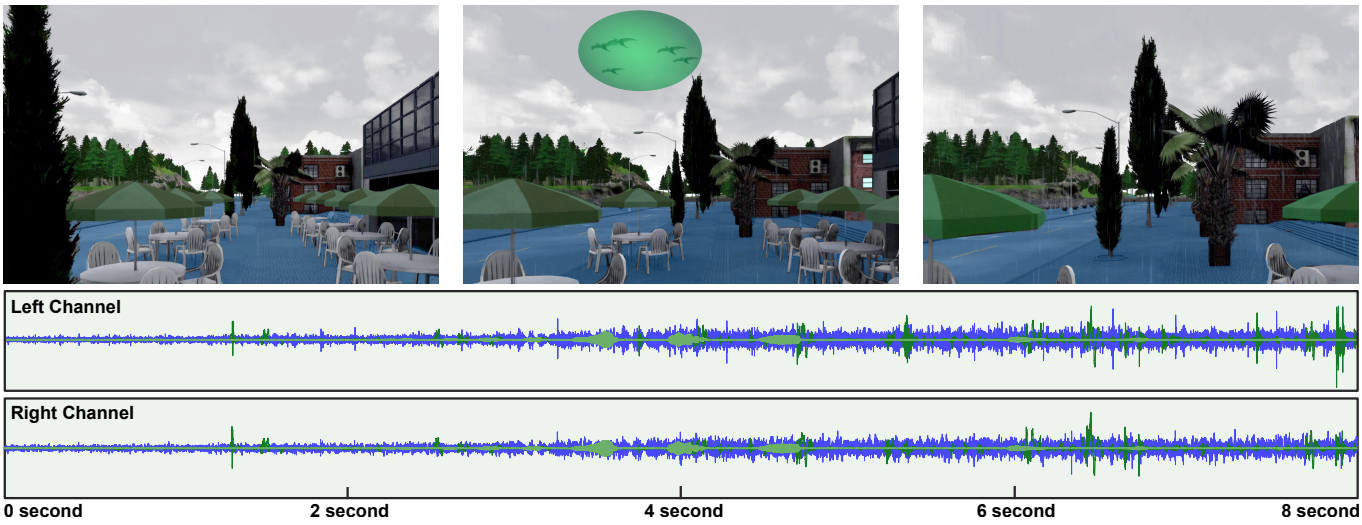


Fig. 1. Our method renders a decorative sound texture of a city street during a rainstorm. The images (top row) show the virtual scene from the listener's perspective over an eight second time period. The plots (bottom row) show the corresponding color-coded waveform of the rendered decorative sound texture in the left and right ears. Raindrops hitting the road (blue) is the background texture, raindrops hitting the umbrella (dark green) is the first foreground sound, and birds chirping (light green) is the second foreground sound. All the foreground sounds and background textures were extracted from recordings at Font *et al.* [2013]. The intensity of the background texture increases throughout the eight seconds, as intended by the scene designer. Additionally, the event frequency of the foreground sounds increases over time, which is also controlled by our methods. This scene is built in CARLA [Dosovitskiy et al. 2017].

Audio recordings contain rich information about sound sources and their properties such as the location, loudness, and frequency of events. One prevalent component in sound recordings is the sound texture, which contains a massive number of events. In such a texture, there can be some distinct and repeated sounds that we term as a foreground sound. Birds chirping in the wind is one such decorative sound texture with the chirping as a foreground sound and the wind as a background texture. To render these decorative sound textures in real-time and with high quality, we create two-layer Markov Models to enable smooth transitions from sound grain to sound grain and propose a hierarchical scheme to generate Head-Related Transfer Function filters for localization cues of sounds represented as area/volume sources. Moreover, during the synthesis stage, we provide control over the frequency and intensity of sounds for customization. Lastly, foreground sounds are often blended into background textures such as the sound of rain splats on car surfaces becoming submerged in the background rain. We develop an extraction component that outperforms existing learning-based methods to facilitate our synthesis with perceptible foreground sounds and well-defined textures.

CCS Concepts: • **Computing methodologies** → *Virtual reality*.

Additional Key Words and Phrases: Soundscape, sound texture, sound rendering, sound synthesis, sound auralization, and sound extraction.

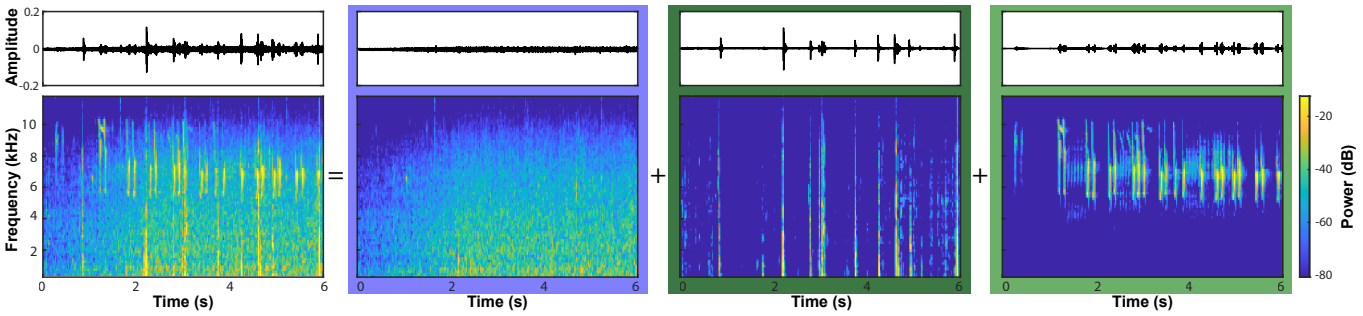
ACM Reference Format:
Jinta Zheng, Shih-Hsuan Hung, Kyle Hiebel, and Yue Zhang. 2020. Real-Time Rendering of Decorative Sound Textures for Soundscapes. *ACM Trans. Graph.* 39, 6, Article 271 (December 2020), 12 pages. https://doi.org/10.1145/3414685.3417875

Authors' addresses: Jinta Zheng, Oregon State University, USA, zhengjinta@outlook.com; Shih-Hsuan Hung, Oregon State University, USA, hungsh@oregonstate.edu; Kyle Hiebel, Oregon State University, USA, hiebelky@oregonstate.edu; Yue Zhang, Oregon State University, USA, zhangyue@oregonstate.edu.

0730-0301/2020/12-ART271 $15.00
https://doi.org/10.1145/3414685.3417875

1 INTRODUCTION

Soundscapes describe the acoustic environment of large scenes and convey messages about the environment to listeners. Soundscapes containing sounds such as babies crying, trees rustling, birds chirping, and fire burning can contribute to a listener's perception of environmental changes or immediate danger. Consequently, 360 videos, virtual reality (VR) scenes [Begault and Trejo 2000; Härmä



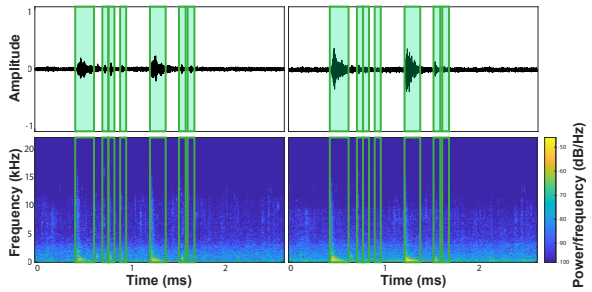
(a) A rainy day on a street (b) Rain dropping on the ground (c) Rain dropping on umbrellas (d) Birds chirping nearby

Fig. 2. An example of a decorative sound texture with waveform (top row) and spectrogram (bottom row). (a) A recording of a rainy day on a street consists of (b) rain dropping on the ground as a background texture, indicated by the light blue border, (c) rain dropping on umbrellas as a foreground sound, indicated by the dark green border, and (d) birds chirping as another foreground sound, indicated by the light green border. The recording is from Font *et al.* [2013].

et al. 2004; Shin et al. 2019], and movie scene designs seek real-time sound synthesis with an emphasis on realistic human perception. In addition, sound extension and augmentation of data from surveillance videos for autonomous cars, security, and specific sound collections require soundscape design [McFee et al. 2018; McLoughlin et al. 2015].

Sound textures, which are compositions of many similar random events, have previously been studied to create real-time and high quality synthesis [Saint-Arnaud and Popat 1995; Schwarz 2011]. Much work has been proposed for rendering sound textures with sound texture synthesis [Heittola et al. 2014; McDermott and Simoncelli 2011; Verron et al. 2009] and auralization [Schissler et al. 2016; Zhang et al. 2018, 2019]. However, a recording of a soundscape concurrently captures a mixture of sound textures as well as some distinct sounds [Zhu and Wyse 2004]. For example, in the scenario shown in Fig. 1, some raindrops hit the ground, but some raindrops hit other material such as umbrellas or cars making a different sound. If we render all raindrops as a single sound texture, the spatial information of dynamic objects in the soundscape, such as umbrellas or cars, will be ignored, creating inconsistency with the expected human perception.

We define decorative sound textures in which *foreground sound*s coexist with *background texture*s as shown in Fig. 2. The sound of raindrops hitting the ground is a background texture, while the sound of raindrops hitting umbrellas, and birds chirping are foreground sounds. Background textures are homogeneous and stationary, so they provide constant information over a period of time [Kell and McDermott 2019]. On the other hand, foreground sounds are collections of repeated sounds which have similar spectrogram patterns and clear boundaries between sounds. They also have a distinct frequency distribution [Panda and Srikanthan 2011] and often have a higher sound pressure level than background textures do. Furthermore, based on our observation of the audio recordings, foreground sounds vary more in temporal frequency over time, shown in Fig. 3, and expose stronger localization cues, i.e., the amplitudes are different for the left and the right ears.



(a) Left Channel (b) Right Channel

Fig. 3. Background textures have a homogeneous difference in loudness to the (a) left ear and (b) right ear; while the foreground sound events (green boxes) have varying differences. In this example, the background texture is rain drops hitting the ground and the foreground sound is heavy rain drops hitting a metal roof. The recording is from Font *et al.* [2013].

As a supplementary pre-processing step, we provide an extraction method which enables users to extract foreground sounds from a background texture. This method uses a 1-D mask in the time domain of the spectrogram and preserves the background texture with minimal distortion. After extraction, foreground sounds and background textures can be attached individually to area/volume sources in virtual scenes.

To render decorative sound textures in real-time, we propose a granular sound synthesis method with a Markov model and an auralization method with Head-Related Transfer Functions (HRTFs) that render foreground sounds and background textures robustly while focusing on realistic human perception. Due to the different properties of foreground sounds and background textures, we develop different schemes to render the sounds respectively. To synthesize decorative sound textures, we propose a two-layer Markov model. The first layer captures the dependency of foreground sound events and the changes in background texture entropy, while the second layer reproduces smooth transitions using a Markov chain. The two-layer Markov models allow the temporal frequency of foreground sounds and the entropy of the resulting background textures to be controlled, following events in the virtual scene. In order to efficiently auralize decorative sound textures from large and dynamic area/volume sources, we employ a hierarchical grid structure to encode HRTFs from an HRTF database on 2D interaural-polar coordinates. Since foreground sounds can indicate isolated positions of some objects, we sample points in the area/volume sources based on the listener's perception and render the sound from the

sampled points. For background textures, we render sound from the entire area/volume sources by summing the pre-integrated HRTFs in hierarchical grid cells that contribute to the sources. We mix the resulting foreground sounds and background textures for the final decorative sound textures for the listeners.

The major contributions of our work are:

- A practical approach to extract foreground sounds from background textures.
- A two-layer Markov model for real-time sound synthesis that captures the different properties of foreground and background components and allows user control of these properties at run-time, and
- An efficient HRTF-based auralization with a hierarchical grid scheme and location sampling for dynamic area/volume sources in free field.

We demonstrate that our decorative sound texture rendering technique generates realistic real-time audio following events in virtual scenes with Unreal Engine 4^{TM}. We conduct experiments to show the high reliability of our decorative sound texture synthesis and the efficiency of our HRTF-based auralization with the hierarchical grid. Furthermore, we conduct a perception evaluation with user studies of our synthesis and auralization.

2 RELATED WORKS

There is much work in the area of soundscape rendering with *sound synthesis*, *sound auralization*, and *sound propagation*. We summarize some of the work most closely related to ours in the following subsections.

2.1 Sound Synthesis

A set of synthesis methods has been developed for a wide range of sounds. Much effort is also seen in real-time synthesis. There are three dominant categories of methods based on their strategies: *granular synthesis*, *filter-based synthesis*, and *physically-based synthesis*.

Granular Synthesis [Roads 1988] is a common scheme for sound synthesis which splits input recordings into short audio clips, called grains, and reorders the grains to create new sounds. Much work exists on this concept, and they differ in how they conduct the sound texture analysis and which metrics they use to guide the grain combinations [Heittola et al. 2014; O'Leary and Roebel 2014; O'Leary and Röbel 2016; Schwarz 2011]. All of the aforementioned work focuses on offline sound synthesis. Schwarz and Caramiaux [2013] present a real-time sound texture synthesis approach using a similarity metric based on descriptor choice for sound texture grains, but this work is not suitable for foreground sounds.

The idea of using Markov models for synthesis can be found in music synthesis [Van Der Merwe and Schulze 2010] to ensure natural transition between unit segments. Another approach, termed lapped texture synthesis in computer graphics [Praun et al. 2000], also employs the idea of assembling similar grains or patterns but without blending between overlapping patches. This technique works best for textures that have clear sound boundaries and are homogeneous. The example-based synthesis method proposed by Wenger and Magnor [2011] segments the grains at the correct location of audios and interchanges similar grains based on an error matrix for Markov models in real-time. They further employed a multi-resolution scheme to reduce the computational cost and local repetitions of sounds. However, for foreground sounds, considering the similarity is insufficient since the time lapse and dependency of the sound events are also important. Furthermore, due to the global structure of background textures, such as wind from steady to gusty and tidal ocean waves, an incorrect segmentation can make transition of resulting sounds abrupt and unnatural to listeners. Our motivation is to achieve real-time synthesis for decorative sound textures and to control the synthesized results. We propose a granular synthesis method with a two-layer Markov model that can reproduce the transition and dependency of foreground sound events and the global structure of background textures.

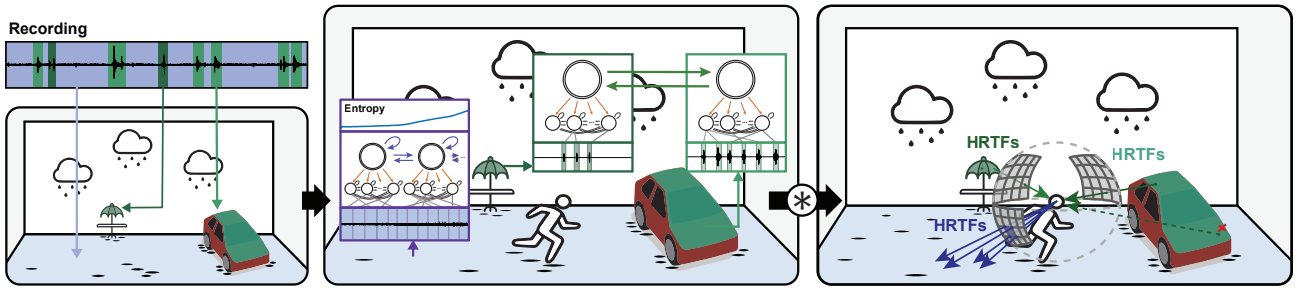
McDermott and Simoncelli [2011] proposed a filter-based synthesis method that utilized noise signals for sound texture synthesis by measuring statistics on the noise. They presented a statistical model based on temporal and spectral correlation of frequency sub-bands in a given sound texture. With the statistical model, McDermott and Simoncelli developed a filtering process on noise to reproduce sound textures. Liao *et al.* [2013] and Bruna and Mallat [2013] extended this concept by including different statistical models for better sound texture synthesis. However, since noise is a continuous signal, the filter-based synthesis falls short when synthesizing sounds with a sparse distribution (i.e., foreground sounds). Additionally, these filter-based synthesis methods require long computation time to synthesize audio. They would not be applicable for real-time rendering.

For a more realistic sound, some work employs physical models to guide the generation of sounds. A practical framework for physically-based synthesis is presented by Wang *et al.* [Wang et al. 2018] with a wide variety of physical simulation models. Some methods are tailored to a particular type of sound such as rain [Liu et al. 2019], fire [Chadwick and James 2011], or fluids [Zheng and James 2009] for high quality synthesis. These physically-based synthesis models can reproduce the complex physical phenomena of nature sounds in virtual scenes, but due to the heavy numerical computation, these sounds are often rendered offline for animations.

2.2 Sound Auralization

There is much existing research that addresses how to effectively deliver the spatial information of acoustic events to listeners in virtual environments. The listeners rely on the following cues for sounds in space: *interaural level differences* (ILD which measures the difference in level and frequency distribution of a sound between the two ears), *interaural time differences* (ITD which measures the difference in time of a sound arriving at each ear), and *directional transfer function* (DTF which measures how a listener's head, ear canal, pinna, and torso change the intensity of sounds with different frequencies at each ear). Using HRTF is one common technique to capture sound propagation from a specific position to the listener's ears, and to facilitate realistically describing the changes of ILD, ITD and DTF due to the listeners' movement.

In virtual scenes, binaural rendering is reproduced by performing convolution of the audio signals with HRTFs. For compression and



(a) Decorative Sound Texture Processing (b) Decorative Sound Texture Synthesis (c) Decorative Sound Texture Auralization

Fig. 4. Rendering of a decorative sound texture includes sound synthesis and auralization. (a) We provide a foreground sound extraction algorithm to create a decorative sound texture with foreground sounds and a background texture on area/volume sources in virtual scenes. (b) At run-time, for decorative sound texture, we synthesize the foreground sounds (green) and background texture (blue) with different Markov model designs. (c) For decorative sound texture auralization, we compute the convolution of the synthesized sound and HRTF filters constructed with hierarchical grids to efficiently capture the location information for the area/volume sources of foreground sounds and background textures. We mix the auralized foreground sound and background textures to generate the final sound for the listener.

computational efficiency, there are two representations of HRTFs: the spherical-harmonics (SH) HRTFs [Evans et al. 1998; Rafaely and Avni 2010; Romigh et al. 2015] and HRTFs by interpolation methods [Begault and Trejo 2000; Freeland et al. 2002; Gamper 2013]. Although projecting the measured samples onto a SH basis can be computationally efficient, SH representations might distort the measured HRTFs [Rafaely and Avni 2010]. Moreover, the given SH order limits the maximum frequency that can be accurately represented [Romigh et al. 2015]. Schissler *et al.* [2016] use an area/volume source projection based on Monte Carlo methods and SH HRTFs to render area/volume sources. In this case, the number of rays used in ray-intersection testing with the Monte Carlo method becomes a factor that influences performance. In contrast, our HRTF auralization is based on interpolation representation of HRTF samples using the full resolution of the database. We sample the area/volume source on a grid space and accelerate the HRTF construction with a hierarchical grid which helps reduce the number of the ray-intersection tests for background textures.

2.3 Sound Propagation

We categorize the previous work of sound propagation into *numerical acoustics* [Raghuvanshi et al. 2009; Raghuvanshi and Snyder 2018] and *geometrical acoustics* [Cao et al. 2016; Schissler et al. 2014]. The numerical acoustics methods solve the wave equation to create a precise sound propagation, while the geometrical acoustics methods utilize rays to efficiently approximate the sound propagation. For ambient sounds, Zhang *et al.* [2018; 2019] simulated sound fields with incoherent sources by encoding the resulting power over the direction, the position in scenes, and the rapidity of received events for real-time computation. Our work currently focuses on sound synthesis and sound auralization for decorative sound textures in free field, and improving this aspect is part of our immediate future work.

3 OUR METHOD

Our system contains three different stages: i) decorative sound texture pre-processing, ii) synthesis, and iii) auralization. In the first stage, we create decorative sound textures in virtual scenes, either from existing foreground sounds and background textures, or from a recording that we apply our extraction algorithm on. For rendering, foreground sounds and background textures are attached to area/volume sources of triangular meshes and spheres in the scene. In the second stage, at run-time, foreground sounds and background textures are synthesized with pre-built two-layer Markov models. In the last stage, we auralize the synthesized foreground sounds and background textures in area/volume sources through the convolutions with constructed HRTFs. Our method is depicted in Fig. 4 and we describe the involved details next.

3.1 Decorative Sound Texture Extraction

A decorative sound texture consists of arbitrary foreground sounds and a background texture from recordings. However, a recording typically includes multiple sound sources and types, and obtaining modular components of the recording can maximize the use of each component in a soundscape. For example, birds chirping in the wind can be segmented into two modular components: bird chirping and wind blowing. Furthermore, we believe that foreground sounds are dependent on the background texture, i.e., birds chirping in the wind is different from birds chirping in a windless environment. To create decorative sound textures from a recorded mix of foreground sounds and background textures, we propose an algorithm to extract foreground sounds while preserving the background texture from the recording.

We build our extraction algorithm based on the probabilistic latent variable model (PLVM) [Bryan and Mysore 2013; Smaragdis et al. 2006]. The PLVM models the sound spectrum as distributions with the probabilistic latent component analysis and extracts sets of distinct sounds that are sparsely distributed in audios. We treat the spectrogram of a given recording as a linear combination of spectral components over time. Since the recording may contain multiple foreground sounds, we reject unwanted time frames through a 1-D mask. This rejection mask guarantees that our algorithm can find the target foreground sound even when it overlaps with the other sounds. By recursively extracting the foreground sounds, we obtain the background texture of the recording, which is the remaining sound.

However, it can be difficult for PLVM to extract sounds from noise such as a background texture, so we propose a two-stage extraction method. These two stages estimate different penalties based on the property of decorative sound textures, where the background texture is locally stationary and quieter than the foreground sounds. The first stage locates the potential target foreground sound and the second stage computes the probability that the residual portion is still the target foreground sound. The penalty represents the posterior probability that a frame is the target foreground sound. A penalty equal to 1.0 implies the frame is the target foreground sound. A penalty equal to 0.5 is the default setting of PLVM for unsupervised learning.

To estimate the penalty, we first transform the audio signal to the Mel-spectrogram that weighs frequency bands on the spectrogram based on human perception [Rabiner and Schafer 2011]. We then measure the overlap between the target signal and residual signal to compute the penalty. When a sound is distinguishable from the other sounds, it implies that this sound has a higher intensity or it occurs at a different frequency band. Therefore, for two time frames i and j in the Mel-spectrogram, the overlap measure is performed with the spectral power in time-frequency bins as

$$\eta(e_i, e_j) = \frac{\sum_f \min(e_i(f), e_j(f))}{\sum_f e_i(f)}, \tag{1}$$

where $e_i(f)$ is the Mel-spectral power of the frequency band f for time frame i and $\eta(e_i, e_j)$ indicates the percentage of e_i that's inside e_j.

For the first stage, we collect the time frames outside the 1-D rejection mask, R, as the potential target foreground sound. Since the background texture is locally stationary, we segment the continuous time frames in R into clips to maintain the local features of the sound. For each frequency band, f, we gather the maximum Mel-spectral power, $e_{max}(f)$ and the penalties, $Pe(f, i)$. For each time frame, i is estimated as

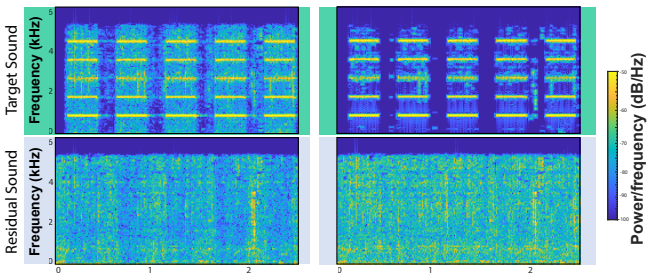
$$Pe(f, i) = \begin{cases} 0.0 & \text{if } \eta(e_i, e_{max}) > 0.98 \\ 0.5 & \text{otherwise} \end{cases}. \tag{2}$$

We then obtain the potential frames, T, of the target foreground sound. The results based on the first stage penalty are shown in Fig. 5 (a).

In the second stage, shown in Fig. 5 (b), we further separate the target foreground sound where the background texture remains in the potential frames, T, of the first stage. For each frequency band, f, to prevent over extraction of the foreground sound, we calculate the minimum percentage of the 1-D rejection mask, R, that resides in T. We then estimate the penalties $Pe(f, t)$ for the time frame $t \in T$ as

$$Pe(f, t) = \begin{cases} \min(1.0 - \frac{e'_{min}(f)}{e'_t(f)}, 0.5) & \text{if } \eta(e_{min}, e_t) > 0.98 \\ 0.5 & \text{otherwise} \end{cases}, \tag{3}$$

where $e_t(f)$ is the Mel-spectral power and $e_{min}(f)$ is the minimum Mel-spectral power in R. The spectral powers $e'_{min}(f)$ and $e'_t(f)$ are of the unweighted spectrogram corresponding to $e_{min}(f)$ and $e_t(f)$. To obtain more precise background textures in R, we also constrain the spectral power of the background texture in each time-frequency between $e'_{min}(f)$ and $e'_{max}(f)$ after the PLVM extraction.



(a) The First Stage (b) The Second Stage

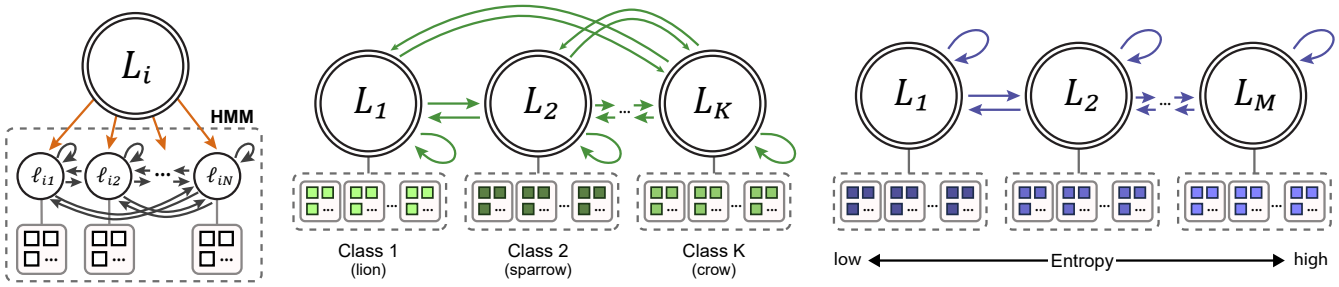
Fig. 5. The resulting target foreground sound and residual sound from PLVM of (a) the first stage and (b) the second stage of extraction.

We evaluate the accuracy of our decorative sound texture extraction and compare to state-of-the-art approaches in Section 5.1.

3.2 Decorative Sound Texture Synthesis with Two-layer Markov Model

To achieve real-time synthesis of decorative sound textures, we adopt the Hidden Markov Model (HMM) which captures the characteristics of events embedded in the extracted grains to retain smooth transitions and natural event distribution. Instead of learning the hidden state of the HMM using a statistical approach, we design the hidden states based on k-means clustering of the grains. We segment grains of foreground sounds and background textures differently. A grain from a foreground sound is one sound event with a discrete beginning and end, such as one bird chirp; while a grain from a background texture has uniform length and contains multiple events, such as many raindrops hitting the ground. Moreover, we develop the two-layer Markov model that encodes the *a priori* information of decorative sound textures in the first layer. Specifically, for foreground sounds, the *a priori*-class layer captures the transition probability between distinct classes of sounds, such as dogs barking and birds chirping. For background textures, the *a priori*-class layer is constructed by entropy analysis, which guarantees the changes in the resulting background texture will be either completely random or random with some repeated structures. Therefore, we can encode background textures such as wind changing from steady to gusty, and ocean waves advancing and receding. Fig. 6 depicts the structure of our two-layer model and the *a priori*-class of foreground sounds and background textures.

To construct the two-layer structure of Fig. 6 (a), we first take the grains, $\mathcal{G}_i$, in an *a priori*-class of the decorative sound texture and create an *a priori*-class state in the first layer. For the second layer of the HMM, we utilize the k-means algorithm to cluster $\mathcal{G}_i$ into N subsets, $\{g_{ij}\}_{j=1}^{N}$, based on the MFCCs [Muda et al. 2010] which estimate the similarity of sounds based on human perception. We then create N hidden states under the *a priori*-class state for the HMM layer and attach g_{ij} to the hidden states. Next, we connect the *a priori*-class state to each of the hidden states with emission probabilities (orange arrows in Fig. 6(a)) and the hidden states to each other with transition probabilities (black arrows in Fig. 6(a)).



(a) Two-layer Structure (b) Foreground Sounds (c) Background Textures

Fig. 6. A two-layer Markov model consists of (a) structures where an *a priori*-class state (double circle) links to hidden states (single circles) that contain clusters of grains (boxes). We first compute the emission probability from the *a priori*-class state (orange arrows) and, for the HMM, we calculate the transitions of the hidden states (black arrows) based on the k-means clustering of the grains (rounded rectangle). (b) For foreground sounds, we create the first layer of the model by connecting the *a priori*-class states with all others based on the dependency between the class (rounded rectangle with dashed line). (c) For a background texture, we build up the first layer of the model by connecting the *a priori*-class states with their adjacent *a priori*-class states based on the entropy of the grains in the classes (rounded rectangle with dashed line).

First, the emission probability from the *a priori*-class state, L_i, to hidden states, ℓ_{ij}, is calculated as

$$P_r(\ell_{ij}|L_i) = \frac{|g_{ij}|}{|\mathcal{G}_i|}, \tag{4}$$

where $|g_{ij}|$ and $|\mathcal{G}_i|$ are the number of the grains in g_{ij} and $\mathcal{G}_i$. Second, the transition probability from ℓ_{ij} to ℓ_{ik} is calculated as

$$P_r(\ell_{ik}|\ell_{ij}) = \frac{C(\ell_{ij}, \ell_{ik})}{|g_{ij}|}, \tag{5}$$

where $C(\ell_{ij}, \ell_{ik})$ is the number of grains in g_{ij} which come before a grain in g_{ik} in the original recording. The emission and transition probabilities allow us to recreate the distributions and dependencies of grains in the original recording.

At run-time, our algorithm samples grains in the layers according to the current *a priori*-class state of the *a priori*-class layer, and the current hidden state of the HMM layer. We develop two rules for sampling:

(1) When the current *a priori*-class state enters the *a priori*-class state L_i, our algorithm will randomly set the current hidden state to ℓ_{ij} based on the emission probabilities, $P_r(\ell_{ij}|L_i)$, where $j = 1, 2, ..., N$.
(2) If the current *a priori*-class state stays in L_i, our algorithm will randomly move the current hidden state from the hidden state, ℓ_{ij}, to another hidden state, ℓ_{ik}, according to the transition probabilities, $P_r(\ell_{ik}|\ell_{ij})$, where $k = 1, 2, ..., N$.

After obtaining the hidden state, our algorithm then uniformly samples a grain from the cluster of the hidden state, and appends the grain to the output audio sequence. However, since foreground sounds and background textures have very different properties and usages, we propose different schemes to compose the basic structures of two-layer Markov models. We describe the details in the following subsections.

3.2.1 Two-layer Markov Model for Foreground Sounds. To construct the model, we treat each given class of the foreground sounds as an *a priori* class and create an *a priori*-class state. First, we segment the input audio of foreground sounds into grains of individual events, which is similar to onset detection [Bello et al. 2005; Davis and Agrawala 2018]. We exploit the Mel-spectrogram that downsamples frequency bands based on human perception; otherwise, the comparison will not be applicable for high frequencies where the audio can have very small energy. For each time frame, if the Mel-spectral powers of the input audio is larger than the background texture in one of the frequency bands, we consider this frame as a part of the foreground sound and we collect the continuous frames as the grains. With the grains, we create links between the *a priori*-class states in the *a priori*-class layer in a similar way to the HMM layer and we additionally compute the transition probability of the *a priori*-class states.

In the HMM layer, since the foreground sounds are discrete in time, we encode the time lapse between grains in the recording by regressing the lapses according to a Gaussian distribution. From the transition of a hidden state, ℓ_{ij}, to another hidden state, ℓ_{ik}, we search the grains in ℓ_{ij} that are in front of the grains in ℓ_{ik}. We then collect the time lapse between the pairs of grains and compute the mean, $\mu_i(\ell_{ij}, \ell_{ik})$, and variance, $\sigma_i(\ell_{ij}, \ell_{ik})$. Therefore, the time lapse of ℓ_{ij} to ℓ_{ik} is modeled as

$$\delta(\ell_{ij}, \ell_{ik}) = \alpha(\mu_i(\ell_{ij}, \ell_{ik}) + \kappa\sigma_i(\ell_{ij}, \ell_{ik})), \tag{6}$$

where κ is a random variable in $[0, 1]$ and α is a control parameter for modifying the temporal frequency of foreground sounds. For the time lapse of *a priori*-class states, we use the same description, $\delta(L_i, L_{i'})$, by counting the time interval between the grains in L_i and their next grain that belongs to $L_{i'}$.

Our algorithm synthesizes foreground sounds by randomly transitioning between *a priori*-class states of the Markov model at runtime. For each iteration, our algorithm adds the lapse time by generating the random variable κ for transition and then appends a sampled grain to the sequence of foreground sounds. By applying the lapse time, we avoid many grains crowding in the same time period, and excessive overlapping. In this scheme, our foreground sound synthesis with two-layer Markov models can create smooth sounds like the original recording.

3.2.2 Two-layer Markov Model for Background Textures. To construct the model, segmenting the grains of background textures is

very important. When background textures have a stable or monotonous change, the grains segmented at even intervals have little influence on the structure of the texture. However, when background textures are complex with organized randomness, segmenting the grains at even intervals can alter the structure. Therefore, to detect abrupt changes in background textures, we propose a segmentation based on the cumulative entropy [Di Crescenzo and Longobardi 2009]. We split the background texture into short clips at the local minima of the cumulative entropy and group the clips with similar entropy values for *a priori*-class states. To provide user control of the randomness of the background texture, we sort the segments using the Shannon entropy and then construct the *a priori*-class state layer. We segment the clips into grains with equal lengths and build the HMM layer.

Our algorithm synthesizes the background texture by shifting the current *a priori*-class state step-by-step to the desired *a priori*-class state. To prevent rapid changes between *a priori*-class states, we enforce that the transitions of *a priori*-class states must occur after the synthesis routine of the current *a priori*-class state. We apply cross-fading between adjacent grains to avoid unwanted clicking artifacts. We show the controllability of our background texture synthesis with two-layer Markov models by synthesizing a background texture from low to high entropy in Fig. 7. We further provide additional constraints when the transition between states needs to stay in the global time domain. These cases arise for sounds like ocean waves which have cyclic patterns of intensity. We also observe that as the entropy increases, the number of sound events in the synthesized background texture increases, and the structure becomes more random. To sample a grain under the *a priori*-class state and also keep the pattern, we constrain the transition probabilities of the Markov model to only allow transitions between different hidden states according to the order presented in the recording.

With the two-layer Markov models, we generate realistic decorative sound textures with similar features to the original recordings. We present our evaluations of the realism and quality of our decorative sound texture synthesis in Section 5.2.

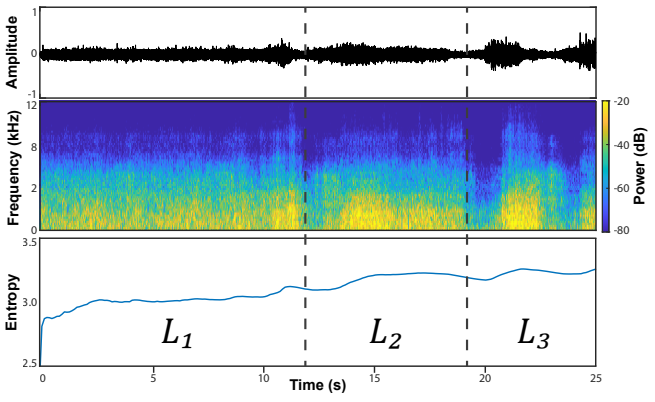


Fig. 7. A synthesized ocean wave sound starting from an *a priori*-class state of low entropy and moving to a high entropy state. The resulting background texture smoothly follows the entropy changes controlled by the *a priori*-class layer of the two-layer Markov model.

3.3 Decorative Sound Texture Auralization

For rendering decorative sound textures, we represent the sound sources as area/volume sources using triangular meshes or spheres and employ an HRTF to auralize the dynamic sources in free field. Generally, the HRTF-based rendering auralizes a sound from an arbitrary location, $\mathbf{x}$, to a listener's head, $(\theta_{\mathbf{x}}, \phi_{\mathbf{x}})$, on 2D interaural-polar coordinates for the left and right ears through convoluting an audio signal and some HRTF filters. We construct the HRTF filters from the four nearest HRTF samples of $(\theta_{\mathbf{x}}, \phi_{\mathbf{x}})$ using bilinear interpolation. We denote the filters $h^{L,R}(\theta_{\mathbf{x}}, \phi_{\mathbf{x}}, t)$ in time domain. In our implementation, we compute the process in the frequency domain, which reduces the computational cost of convolution and maintains the ITD when performing bilinear interpolation on the HRTFs.

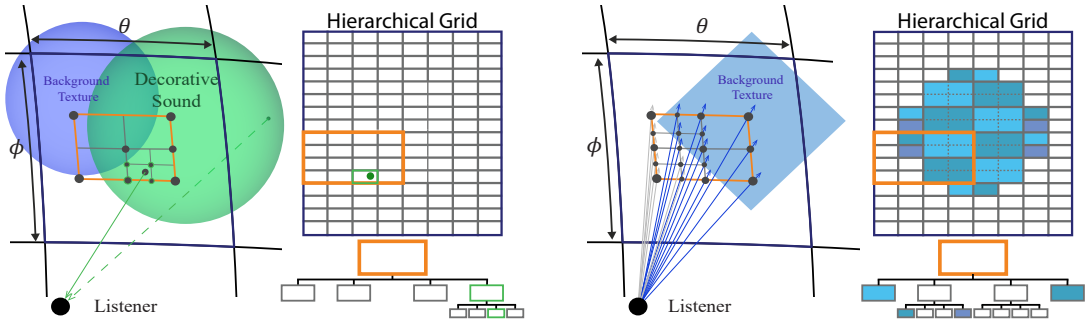
Moreover, we present a pre-computed HRTF with quad-tree structure [Finkel and Bentley 1974] and encode the HRTF database samples with the full resolution to accelerate the construction of HRTF filters. For the quad-tree structure, we build a hierarchical grid by recursively subdividing the grid cells into four quadrants in 2D interaural-polar coordinates. We first augment the HRTF database samples by interpolating the HRTFs at the corner of the leaf cells and then sum up the HRTFs in each child cell for the parent cells. In such a case, a parent cell will contain the HRTF that can be perceived from the area formed by the children cells. In the next subsections, we detail different schemes of auralization of foreground sounds and background textures.

3.3.1 Area/Volume Source for Foreground Sounds. We treat foreground sounds as temporary point sources within the area/volume source for a strong perception of the positions of the objects. For each iteration of a Markov model, a foreground sound, $s_f(t)$, is synthesized with a grain containing a single sound event. To render the individual sound event, we first sample a location, $\mathbf{x}$, in the area/volume source where the sound comes from, shown in Fig. 8 (a). Our system next exploits the HRTF-based rendering to render the sound from the location and adds the time delay τ, which is defined by dividing the distance, $\mathbf{d}_{\mathbf{x}}$, with the sound speed of 343 m/s. The auralization for left and right ears is modeled as

$$p_f^{L,R}(t+\tau) = \frac{1}{\mathbf{d}_{\mathbf{x}}^2} h^{L,R}(\theta_{\mathbf{x}}, \phi_{\mathbf{x}}, t) \circledast s_f(t), \tag{7}$$

where $\circledast$ is the convolution operator.

Moreover, we develop a sampling scheme which chooses locations in the area/volume source for foreground sounds to avoid drowning out by a background texture. Therefore, due to the noise-like properties of background textures, we propose a perception-guided sampling technique using a noise-over-tone masking threshold [Panda and Srikanthan 2011] to select sample locations. Given the foreground sound and a synthesized grain of the background texture, a set of thresholds are estimated by the sound pressure levels (SPL) of the background texture grain in 10 Mel-filter bins. The location of a foreground sound is sampled uniformly in the area/volume source. We utilize these thresholds to compare to the SPLs of the foreground sound to determine which sampled location is audible. We set the maximum iterations to 20 to prevent the sampling routine from entering an infinite loop. If our system rejects all sampled



(a) Foreground Sound (b) Background Texture

Fig. 8. Auralization of area/volume sources of (a) a foreground sound and (b) a background texture. (a) The foreground sound is emitted randomly from a point guided by the perception model that avoids a point (dashed arrow) where the sound is masked by a background texture. The HRTF is interpolated at the point in 2D interaural-polar coordinates using the hierarchical grid. (b) The background texture is heard from the area/volume sources and the HRTF is constructed with the ray-intersection test in the hierarchical grid.

locations, then it will remove the grain of foreground sound from the rendering sequence to save auralization computation time. With this scheme, we render foreground sounds in area/volume sources with random locations for each grain. This provides changes in strength and location of the rendered sounds, in accordance with our observation of the recordings.

3.3.2 Area/Volume Source for Background Textures. For background textures, due to the massive events in the grains of the synthesized sound, it is impossible to trace the distance and time delay for each single event in the grain. Furthermore, the local randomness of a background texture leads to a more consistent difference between left and right ears. We, thus, consider the whole area/volume source as the emission region of the background texture and auralize the synthesized sound, $s_b(t)$, with the HRTFs of the source, $H_b^{L,R}(t)$, as

$$p_b^{L,R}(t) = \frac{1}{\mathbf{d}_s^2} H_b^{L,R}(t) \circledast s_b(t). \tag{8}$$

Note we only apply the distance attenuation from the closest distance, $\mathbf{d}_s$, of the source as the grain of background textures already embeds the distance attenuation and time delay of the individual events [Saint-Arnaud and Popat 1995]. Consequently, our approach includes the location information for background textures and maintains the relative loudness with their foreground sounds.

The HRTF can be constructed with all the points in the area/volume source to the listener in the directions (Θ, Φ),

$$H_b^{L,R}(t) = \frac{1}{Z} \sum_{\theta \in \Theta} \sum_{\phi \in \Phi} h^{L,R}(\theta, \phi, t), \tag{9}$$

where Z is the number of the points. However, the construction with all the points in the area/volume source is impractical. To approximate the integration, our algorithm samples the pre-calculated HRTFs in the hierarchical grid, by ray-intersection tests for triangular meshes and spheres of the area/volume source. With the hierarchical grid, the intersection test starts from the root cell and moves to the leaves, stopping when all the rays from the listener's head to the corners of a cell intersect with the area/volume source (see Fig. 8 (b)). We formulate the process of the HRTFs construction as

$$\widetilde{H}_b^{L,R}(t) = \frac{1}{Z_G} \sum_{g \in G} z(g)\, h^{L,R}(\theta_g, \phi_g, t), \tag{10}$$

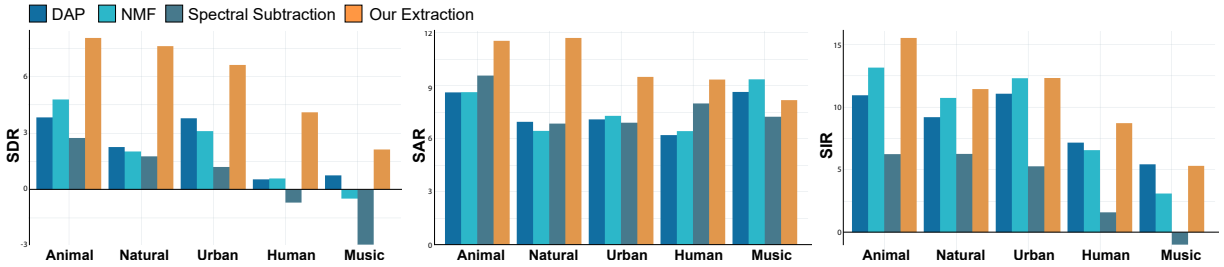
where G is the set of intersected grid cells. We give a weight, $z(g)$, based on the area of the cell g in 2D interaural-polar coordinates and normalize $\widetilde{H}_b^{L,R}(t)$ with Z_G, the sum of the $z(g)$ for $\forall g \in G$. We verify the performance and evaluate the human perception of our results in Section 5.3.

4 IMPLEMENTATION AND PERFORMANCE

Our decorative sound texture rendering is implemented as a plugin for Unreal Engine 4™. All recordings used in our paper are available publicly through the Freesound project [Font et al. 2013]. Please see the accompanying video for details on two virtual scenes, rain in the city and picnic in the park, using decorative sound textures. Here, we discuss the implementation and performance of our system. Our system is evaluated on a computer with an i7-8700K @3.70GHz CPU and an NVIDIA GeForce GTX 2080 GPU.

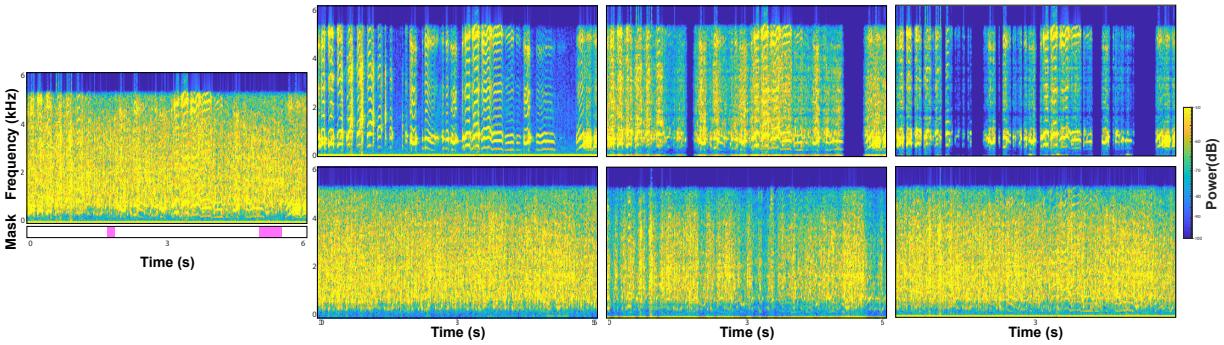
In the decorative sound texture processing stage, we set 100 bases for PLVM to extract the sounds and compute the Mel spectrogram with 70 Mel bands with window size of 2048 samples and 1024 samples overlap. Our extraction takes around 2.5 seconds to detect and separate a foreground sound from a 10 seconds long audio clip at 44100 Hz.

In the synthesis stage, for background textures, our entropy analysis captures the local structure of sound textures over a long period with a window size of 0.14 seconds and an analysis length of 18 seconds. Additionally, we segment the grains with a length of 0.14 seconds. For a 10 seconds recording, our system takes around 7 seconds to build the model of a background texture and around 0.8 seconds for a foreground sound. The computation time of the construction depends on the number of classes in the k-means algorithm and the length of recording. We build up the HMM layer by clustering the grains into N groups and we currently set N to an ad hoc value, 5. At run-time, our system synthesizes 44100 samples (1 second) of a background texture or a foreground sound within 5 milliseconds.



(a) Evaluation of SDR (b) Evaluation of SAR (c) Evaluation of SIR

Fig. 9. Comparison of our decorative sound texture extraction to DAP [Tian et al. 2019], NMF [Spiertz and Gnann 2009], and spectral subtraction [Boll 1979] using (a) SDR, (b) SAR, and (c) SIR. We average the values of SDR, SAR, and SIR over the 5 categories (animal, natural, urban, human, and music). Our extraction results have higher average scores compared to the other methods. Our test database and results can be accessed here: https://github.com/hiebelky/JKSound-Benchmark.



(a) Input and annotations (b) Ground truth (c) Bryan *et al.* (d) Our result

Fig. 10. Comparison of our decorative sound texture extraction to Bryan *et al.* [2013] on (a) a baby laughing in a background texture using the same 1-D mask (purple bars). The top row of (b) – (d) shows the extracted foreground sounds, and the bottom row shows the remaining background texture. (c) The result of Bryan *et al.* [2013] has an SDR of 6.52 dB for the foreground sound, and an SDR of 1.55 dB for the background texture compared to the ground truth. (d) Our result is better with an SDR of 9.08 dB for the foreground sound, and an SDR of 3.06 dB for the background texture.

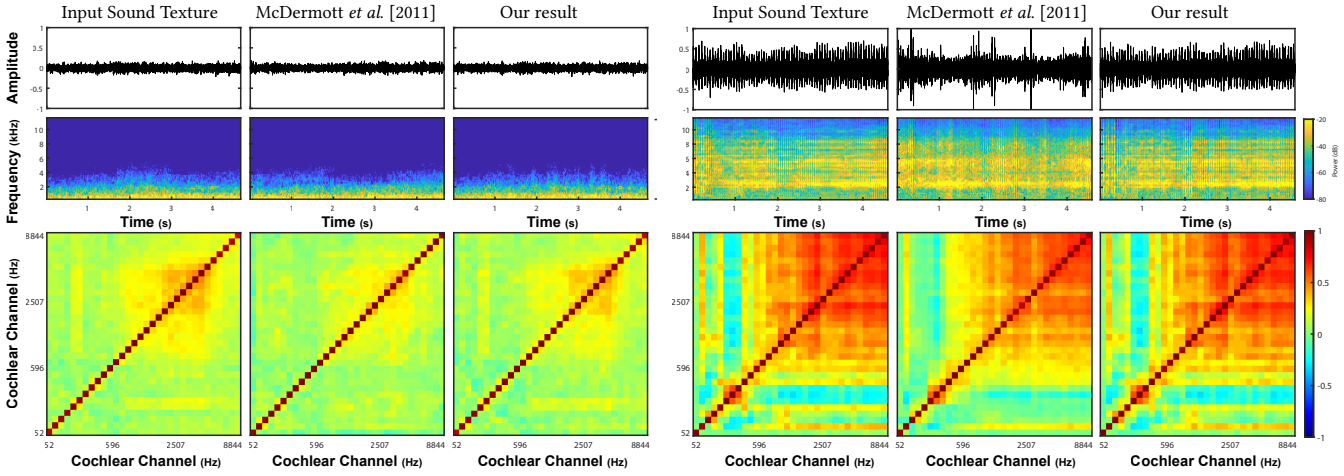
In the auralization stage, for HRTFs, our system uses the CIPIC database [Algazi et al. 2001] which contains 1250 sampled head-related impulse responses (HRIRs). We convert HRIRs to HRTFs using the fast Fourier transform and store them over 200 frequency bands. For the hierarchical grid, we create 4096 leaf cells with 5.626° width and 2.8125° height, which are close to the minimum audible angle threshold, 3.65° [Perrott and Saberi 1990]. Our system builds the hierarchical grid with depth 6 in 0.23 milliseconds. At run-time, our system processes 1024 samples of each source sound sequence for every update. Our system first applies a modified discrete cosine transform on the samples and computes the convolution with the HRTF filters. The binaural sound is then generated by the inverse Fourier transform and overlapped with the last 512 samples from the previous update. This convolution step takes around 0.1 milliseconds. To obtain the HRTFs from a point, we search the grid cell that contains the point on the 2D interaural-polar coordinate and interpolate the HRTFs with HRTF filters stored in the four corners of the cell. The searching and interpolation of HRTFs cost around 0.006 milliseconds. The querying in the hierarchical grid has time complexity $O(log\mathbf{N})$, where $\mathbf{N}$ is the number of cells, 4096. For background textures, we conduct performance experiments in Section 5.3 for area/volume sources using different shapes.

5 RESULTS AND DISCUSSION

In this section, we evaluate our extraction, synthesis, and auralization methods and compare our results to the previous work.

5.1 Evaluation of Decorative Sound Texture Extraction

To evaluate the accuracy of our decorative sound texture extraction, we examine our approach on a subset of the Universal-150 audio benchmark [Tian et al. 2019] with cases from 5 categories (animals, natural sounds, urban sounds, human, and music). We choose 98 cases that contain at least one period when the sounds are not overlapping for the 1-D rejection mask. We use the metrics of i) Signal-to-Artifact Ratio (SAR) which measures artifacts that have been introduced by the separation process, ii) Signal-to-Interference Ratio (SIR), which measures suppression of the unwanted source, and iii) Signal-to-Distortion Ratio (SDR), which is an overall measure that takes into account both SIR and SAR. Higher values for these metrics indicate the separated sounds are cleaner or of higher quality. Fig. 9 shows the metrics averaged within each category in the database and the comparison of our extraction to the state-of-the-art methods: Deep Audio Prior (DAP) [Tian et al. 2019], Non-negative Matrix Factorization (NMF) [Spiertz and Gnann 2009], and spectral subtraction [Boll 1979]. We also compare our method to Bryan *et al.* [2013] which requires 2-D masks on the spectrogram.



(a) Wind blowing (b) Jackhammer

Fig. 11. Synthesized background textures using McDermott *et al.* [2011] (second column) and our method (third column) compared to the original sound texture of (a) wind, and (b) a jackhammer. From top to bottom, the figure shows the waveform, Mel spectrogram, and cross-band envelope correlation matrix for each audio clip. Our resulting background textures reproduce closely the features of the input recordings. The sounds are from Font *et al.* [2013].

In Fig. 10, we use the same 1-D masks on both of the methods to extract the foreground sounds. Compared to the ground truth, our extracted foreground sounds and background texture have higher SDR than those from Bryan *et al.* [2013]. Furthermore, the experiments indicate our extraction with a simple mask performs better when the recording contains a more stationary background texture (in contrast to music).

5.2 Evaluation of Decorative Sound Texture Synthesis

McDermott and Simoncelli [2011] conducted an evaluation on sound texture perception via statistics of the auditory periphery. Hence, we apply their texture statistics to evaluate the reliability of our background texture synthesis. These statistics include marginal moments based on cochlear envelopes and correlations based on modulation bands. Fig. 11 shows the Mel-spectrogram and the cross-band envelope correlation matrix of the input sound textures, results synthesized by McDermott and Simoncelli *et al.* [2011], and results from our method. We further compare both methods' synthesis results to the input sound textures using the root mean square (RMS) errors of the texture statistics. Fig. 12 shows the average RMS errors on 11 sound textures. These RMS errors indicate that our background texture synthesis produces sound textures more similar to the inputs than their method does.

For decorative sound textures, we conduct a perception based evaluation of i) similarity to the input recording, and ii) quality which includes presence of artifacts, such as abrupt loudness or timbral changes, cuts, repetitions, etc. We picked 10 sound recordings of length 4 to 10 seconds; for each recording, we generated 2 sound samples synthesized by our decorative sound texture synthesis and added the original recording as a hidden anchor. We received 12 replies from 20 invited students at our university. Our survey had to be conducted online due to the current arrangement for remote teaching. The students were asked to rate the similarity (from very dissimilar to very similar) and quality (from low to high) of two synthesized sound samples and one hidden anchor, on a scale of 1 to 5. The two sound samples and hidden anchor are ordered randomly to prevent any bias. We employ the analysis of variance (ANOVA) to evaluate our results. For our two samples and the hidden anchor, the p-values are less than 0.05 for both similarity and quality. This indicates that our results are statistically significant. We average the scores of our two sound samples for each recording to get the final result. The average scores for our synthesis are 4.35 for similarity and 4.30 for quality, while for the hidden anchor 4.72 for similarity and 4.75 for quality. From the statistical comparison mentioned earlier and the user study, we conclude that that our decorative sound texture synthesis can closely represent the input recordings and produce high quality audio.

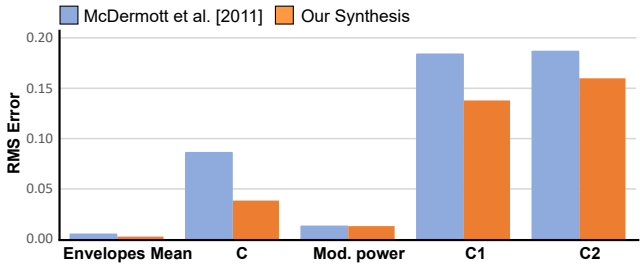


Fig. 12. The RMS errors from comparing our background texture synthesis and the McDermott *et al.* [2011] to the input sound texture over the texture statistics. We evaluate the texture statistics with the mean of the cochlear envelopes of each frequency band (Envelopes mean), the cross-band envelope correlation matrix (C), the modulation power (Mod. power) and two types of modulation correlations (C1: the same modulation frequency but different acoustic frequencies; C2: the same acoustic frequency but different modulation frequencies). The bar chart shows that our resulting background textures have a smaller RMS error from the input recordings.

5.3 Evaluation of Background Texture Auralization

For a background texture, we evaluate the effectiveness and human perception of our hierarchical grid HRTF construction. We compare our method to the HRTF construction without hierarchical grid and to a spherical-harmonic (SH) based method. We implement the SH scheme from the work of Schissler *et al.* [2016], which encodes HRTFs in SH of the 9th order and applies the Monte Carlo projection for area/volume sources. Note the sources made of spheres are only applied with the analytical projection.

For performance testing, we conduct 2 experiments with sources represented as spheres and planes. For volumetric sources, our subjects are spheres of radii increasing from 100 meters to 450 meters in 25 meter increments. For area/volume sources made of triangles, we test the methods using planes of increasing sizes from 100 meters to 1000 meters in 50 meter increments. We measure the cost in time of each method by timing auralization for each sound source over ten thousand iterations. The cost in time includes ray-intersection testing, projections, and filter construction for each method. Fig. 13 illustrates that our HRTF construction using hierarchical grids is only slightly affected by the problem size and is up to around 5.48 times faster than the construction without the hierarchical grids. Fig. 14 shows the averaged cost in time for the two experiments. For the sphere, our method is slightly slower than the SH method using analytical projection. However, for the plane, our method is faster than the SH method using the Monte Carlo projection with 1000 and 10, 000 rays.

To evaluate real-life perception, we compare our background texture auralization to the SH method using 10, 000 rays to ensure a high quality result. To compare just the spatial extent of the sound source in the HRTF construction, we do not apply time delay in either of the methods. We conduct a human perception study in 2 virtual scenes: a beach and a park. For each scene, the listener walks forward and turns. Our videos are 4-10 seconds long using both the SH method and our method. The order of the videos is randomly placed for each scene. Our 10 participants rated the spatial extent (from point-like to expansive) and localization cue (from weak to strong) for the videos. Overall, the average scores of the spatial extent are 3.98 for the SH method and 3.43 for our method. The average scores for the localization cue are 3.85 for the SH method and 4.03 for our method. The p-values are 0.05 for the spatial extent and 0.33 for the localization cue. From this preliminary user evaluation, we observe that our 10 participants feel the sounds auralized by our method have a stronger localization cue but a less expansive spatial extent than the SH method.

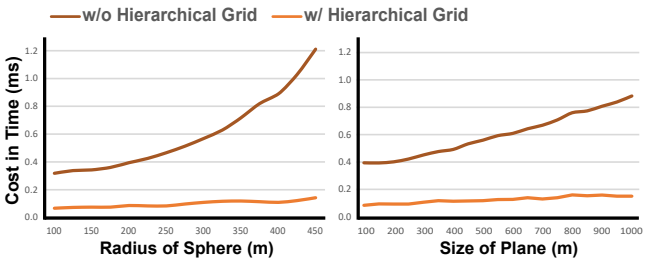
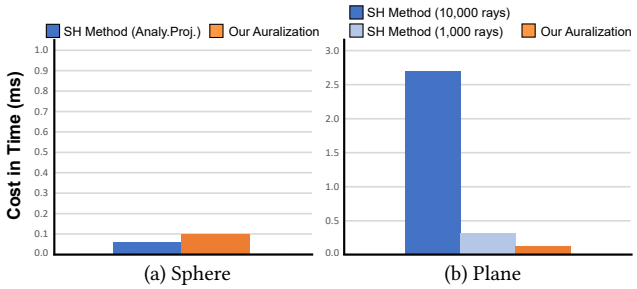


Fig. 13. Comparison of our HRTF construction with and without the hierarchical grid with respect to the size of a sphere or plane. The two experiments indicate our hierarchical grid method improves performance and is slightly dependent on the size of the sound source.



(a) Sphere (b) Plane

Fig. 14. Comparison of our HRTF construction to the SH method based on Schissler *et al.* [2016]. (a) For spheres, our method is slightly slower than the SH method which has only analytical projection. (b) For planes, our method is faster than the SH method using Monte Carlo projection with both 1000 and 10, 000 rays.

6 CONCLUSION AND FUTURE WORK

We present a framework for rendering decorative sound textures that include foreground sounds and homogeneous background textures. Due to the contrasting properties of foreground sounds and background textures, we develop a real-time sound synthesis technique that reproduces decorative sound textures from a recording and provides some control over the temporal frequency of sound events for customization. We also propose a background texture sensitive extraction algorithm to separate foreground sounds and background textures from recordings with a 1-D mask. We demonstrate that our system can reproduce recordings of decorative sound textures and efficiently render a realistic soundscape with dynamic area/volume sources in virtual scenes.

However, our extraction is limited in that it can extract only one foreground sound at a time, and require the foreground sound to have a sparse distribution over the time domain. For our decorative sound texture synthesis, due to the HMM which interchanges grains based on their similarity, our method can not be applied to speech and music with specific rhythm. Another limitation is that the Markov model relies on the k-means algorithm, which requires a set of hidden states. This number of hidden states influences how smooth the final synthesized result is, and the values are chosen heuristically. In addition, dissimilar grains can be clustered due to large variation in amplitudes of background texture. In the future, we plan to look into other algorithms such as graph cut enabled texture segmentation and optimization-based texture synthesis [Kwatra et al. 2005, 2003]. Lastly, our work currently only focuses on free field. Our immediate future work is to enhance sound propagation with physical modeling for dynamic area/volume sources to correctly compute the sound interference in both near-field and far-field, in real-time.

ACKNOWLEDGMENTS

We would like to thank our anonymous reviewers for their valuable comments, and the online participants for their user evaluations.

REFERENCES

V Ralph Algazi, Richard O Duda, Dennis M Thompson, and Carlos Avendano. 2001. The CIPIC HRTF database. In *Applications of Signal Processing to Audio and Acoustics*. IEEE, 2001 IEEE Workshop, 99–102.

Durand R Begault and Leonard J Trejo. 2000. 3-D sound for virtual reality and multimedia. (2000).

Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B Sandler. 2005. A tutorial on onset detection in music signals. *IEEE Transactions on speech and audio processing* 13, 5 (2005), 1035–1047.

Steven Boll. 1979. Suppression of acoustic noise in speech using spectral subtraction. *IEEE Transactions on acoustics, speech, and signal processing* 27, 2 (1979), 113–120.

Joan Bruna and Stéphane Mallat. 2013. Audio texture synthesis with scattering moments. *arXiv preprint arXiv:1311.0407* (2013).

Nicholas Bryan and Gautham Mysore. 2013. An efficient posterior regularized latent variable model for interactive sound source separation. In *International Conference on Machine Learning*. 208–216.

Chunxiao Cao, Zhong Ren, Carl Schissler, Dinesh Manocha, and Kun Zhou. 2016. Interactive sound propagation with bidirectional path tracing. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–11.

Jeffrey N Chadwick and Doug L James. 2011. Animating fire with sound. In *ACM Transactions on Graphics (TOG)*, Vol. 30. ACM, 84.

Abe Davis and Maneesh Agrawala. 2018. Visual Rhythm and Beat. *ACM Trans. Graph.* 37, 4 (2018), 122–1.

Antonio Di Crescenzo and Maria Longobardi. 2009. On cumulative entropies. *Journal of Statistical Planning and Inference* 139, 12 (2009), 4072–4087.

Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An open urban driving simulator. *arXiv preprint arXiv:1711.03938* (2017).

Michael J Evans, James AS Angus, and Anthony I Tew. 1998. Analyzing head-related transfer function measurements using surface spherical harmonics. *The Journal of the Acoustical Society of America* 104, 4 (1998), 2400–2411.

Raphael A. Finkel and Jon Louis Bentley. 1974. Quad trees a data structure for retrieval on composite keys. *Acta informatica* 4, 1 (1974), 1–9.

Frederic Font, Gerard Roma, and Xavier Serra. 2013. Freesound technical demo. In *Proceedings of the 21st ACM international conference on Multimedia*. 411–412.

Fabio P Freeland, Luiz WP Biscainho, and Paulo SR Diniz. 2002. Efficient HRTF interpolation in 3D moving sound. In *Audio Engineering Society Conference: 22nd International Conference: Virtual, Synthetic, and Entertainment Audio*. Audio Engineering Society.

Hannes Gamper. 2013. Head-related transfer function interpolation in azimuth, elevation, and distance. *The Journal of the Acoustical Society of America* 134, 6 (2013), EL547–EL553.

Aki Härmä, Julia Jakka, Miikka Tikander, Matti Karjalainen, Tapio Lokki, Jarmo Hiipakka, and Gaëtan Lorho. 2004. Augmented reality audio for mobile and wearable appliances. *Journal of the Audio Engineering Society* 52, 6 (2004), 618–639.

Toni Heittola, Annamaria Mesaros, Dani Korpi, Antti Eronen, and Tuomas Virtanen. 2014. Method for creating location-specific audio textures. *EURASIP Journal on Audio, Speech, and Music Processing* 2014, 1 (2014), 9.

Alexander JE Kell and Josh H McDermott. 2019. Invariance to background noise as a signature of non-primary auditory cortex. *Nature communications* 10, 1 (2019), 1–11.

Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. 2005. Texture optimization for example-based synthesis. In *ACM SIGGRAPH 2005 Papers*. 795–802.

Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. 2003. Graphcut textures: image and video synthesis using graph cuts. *ACM Transactions on Graphics (ToG)* 22, 3 (2003), 277–286.

Wei-Hsiang Liao, Axel Roebel, and Alvin Su. 2013. On the modeling of sound textures based on the STFT representation. In *Proc. of the 16th Int. Conference on Digital Audio Effects (DAFx-13)*. 33.

Shiguang Liu, Haonan Cheng, and Yiying Tong. 2019. Physically-based statistical simulation of rain sound. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 123.

Josh H McDermott and Eero P Simoncelli. 2011. Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis. *Neuron* 71, 5 (2011), 926–940.

Brian McFee, Justin Salamon, and Juan Pablo Bello. 2018. Adaptive pooling operators for weakly labeled sound event detection. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 26, 11 (2018), 2180–2193.

Ian McLoughlin, Haomin Zhang, Zhipeng Xie, Yan Song, and Wei Xiao. 2015. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23, 3 (2015), 540–552.

Lindasalwa Muda, Mumtaj Begam, and Irraivan Elamvazuthi. 2010. Voice recognition algorithms using mel frequency cepstral coefficient (MFCC) and dynamic time warping (DTW) techniques. *arXiv preprint arXiv:1003.4083* (2010).

Sean O'Leary and Axel Roebel. 2014. A two level montage approach to sound texture synthesis with treatment of unique events.. In *DAFx*. 1–1.

Seán O'Leary and Axel Röbel. 2016. A montage approach to sound texture synthesis. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 24, 6 (2016), 1094–1105.

Ashish Panda and Thambipillai Srikanthan. 2011. Psychoacoustic model compensation for robust speaker verification in environmental noise. *IEEE transactions on audio, speech, and language processing* 20, 3 (2011), 945–953.

David R Perrott and Kourosh Saberi. 1990. Minimum audible angle thresholds for sources varying in both elevation and azimuth. *The Journal of the Acoustical Society of America* 87, 4 (1990), 1728–1731.

Emil Praun, Adam Finkelstein, and Hugues Hoppe. 2000. Lapped textures. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. 465–470.

Lawrence R Rabiner and Ronald W Schafer. 2011. *Theory and applications of digital speech processing*. Vol. 64. Pearson Upper Saddle River, NJ.

Boaz Rafaely and Amir Avni. 2010. Interaural cross correlation in a sound field represented by spherical harmonics. *The Journal of the Acoustical Society of America* 127, 2 (2010), 823–828.

Nikunj Raghuvanshi, Rahul Narain, and Ming C Lin. 2009. Efficient and accurate sound propagation using adaptive rectangular decomposition. *IEEE Transactions on Visualization and Computer Graphics* 15, 5 (2009), 789–801.

Nikunj Raghuvanshi and John Snyder. 2018. Parametric directional coding for precomputed sound propagation. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 108.

Curtis Roads. 1988. Introduction to granular synthesis. *Computer Music Journal* 12, 2 (1988), 11–13.

Griffin D Romigh, Douglas S Brungart, Richard M Stern, and Brian D Simpson. 2015. Efficient real spherical harmonic representation of head-related transfer functions. *IEEE Journal of Selected Topics in Signal Processing* 9, 5 (2015), 921–930.

Nicolas Saint-Arnaud and Kris Popat. 1995. Analysis and synthesis of sound textures. In *in Readings in Computational Auditory Scene Analysis*. Citeseer.

Carl Schissler, Ravish Mehra, and Dinesh Manocha. 2014. High-order diffraction and diffuse reflections for interactive sound propagation in large environments. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–12.

Carl Schissler, Aaron Nicholls, and Ravish Mehra. 2016. Efficient HRTF-based spatial audio for area and volumetric sources. *IEEE transactions on visualization and computer graphics* 22, 4 (2016), 1356–1366.

Diemo Schwarz. 2011. State of the art in sound texture synthesis. In *Digital Audio Effects (DAFx)*. 221–232.

Diemo Schwarz and Baptiste Caramiaux. 2013. Interactive sound texture synthesis through semi-automatic user annotations. In *International Symposium on Computer Music Multidisciplinary Research*. Springer, 372–392.

Mincheol Shin, Stephen W Song, Se Jung Kim, and Frank Biocca. 2019. The effects of 3D sound in a 360-degree live concert video on social presence, parasocial interaction, enjoyment, and intent of financial supportive action. *International Journal of Human-Computer Studies* 126 (2019), 81–93.

Paris Smaragdis, Bhiksha Raj, and Madhusudana Shashanka. 2006. A probabilistic latent variable model for acoustic modeling. (2006).

Martin Spiertz and Volker Gnann. 2009. Source-filter based clustering for monaural blind source separation. In *Proceedings of the 12th International Conference on Digital Audio Effects*.

Yapeng Tian, Chenliang Xu, and Dingzeyu Li. 2019. Deep Audio Prior. *ArXiv* abs/1912.10292 (2019).

Andries Van Der Merwe and Walter Schulze. 2010. Music generation with markov models. *IEEE MultiMedia* 18, 3 (2010), 78–85.

Charles Verron, Mitsuko Aramaki, Richard Kronland-Martinet, and Grégory Pallone. 2009. Spatialized synthesis of noisy environmental sounds. In *Auditory Display*. Springer, 392–407.

Jui-Hsien Wang, Ante Qu, Timothy R Langlois, and Doug L James. 2018. Toward wave-based sound synthesis for computer animation. *ACM Trans. Graph.* 37, 4 (2018), 109–1.

Stephan Wenger and Marcus Magnor. 2011. Constrained example-based audio synthesis. In *2011 IEEE International Conference on Multimedia and Expo*. IEEE, 1–6.

Zechen Zhang, Nikunj Raghuvanshi, John Snyder, and Steve Marschner. 2018. Ambient sound propagation. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–10.

Zechen Zhang, Nikunj Raghuvanshi, John Snyder, and Steve Marschner. 2019. Acoustic texture rendering for extended sources in complex scenes. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 1–9.

Changxi Zheng and Doug L James. 2009. Harmonic fluids. In *ACM Transactions on Graphics (TOG)*, Vol. 28. ACM, 37.

Xinglei Zhu and Lonce Wyse. 2004. Sound texture modeling and time-frequency LPC. In *Proceedings of the 7th international conference on digital audio effects DAFX*, Vol. 4.