Techniques for Soundscape Retrieval and Synthesis

by

Brandon Michael Mechtley

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved November 2013 by the
Graduate Supervisory Committee:

Andreas Spanias, Co-Chair
Hari Sundaram, Co-Chair
Perry Cook
Baoxin Li

ARIZONA STATE UNIVERSITY

December 2013

ABSTRACT

The study of acoustic ecology is concerned with the manner in which life interacts with its environment as mediated through sound. As such, a central focus is that of the *soundscape*: the acoustic environment as perceived by a listener. This dissertation examines the application of several computational tools in the realms of digital signal processing, multimedia information retrieval, and computer music synthesis to the analysis of the soundscape. Namely, these tools include a) an open source software library, *Sirens*, which can be used for the segmentation of long environmental field recordings into individual sonic events and compare these events in terms of acoustic content, b) a graph-based retrieval system that can use these measures of acoustic similarity and measures of semantic similarity using the lexical database *WordNet* to perform both text-based retrieval and automatic annotation of environmental sounds, and c) new techniques for the dynamic, realtime parametric morphing of multiple field recordings, informed by the geographic paths along which they were recorded.

For Muffin.

This dissertation involves topics relevant to multimedia information retrieval and in so doing frequently talks about such things as the "classification," "segmentation," and "discrimination" of "objects." The term "sound" is used quite a bit to refer to short digital PCM audio recordings, and I even have the audacity at one point to quantitatively evaluate what's more or less an aesthetic experience. I'd feel dishonest to not point out, however, that it's become clear to me that in order to truly understand any matter, I don't need to look any further than the immediate presence of felt experience, blooming buzzing confusion, eternal now, unspeakable, or whatever one wishes to call *it* which doesn't need a name, practice, or study. More than an esoteric apology, a necessary component of my understanding of ecology is the intuitive awareness that, in immediate experience, these types of boundaries don't exist. If any of this is valuable, then *listening* is just as important, if not more important than, any computational analysis I could ever hope to perform. I do, of course, enjoy contributing to the "buzzing" part of the confusion.

Before getting into a lengthy list of "thank you"'s, I'd like to mention that this work heavily depends on that of a group formed as part of the School of Arts, Media, and Engineering—primarily that of Harvey Thornburg, Gordon Wichern, Alex Fink, Jiachen Xue, and Jinru Liu. The second chapter, which describes an open source library called *Sirens*, is an implementation of the techniques described in Dr. Wichern's dissertation, which I thoroughly recommend reading for background on event detection and retrieval. Many of the ideas were also developed as a collaboration with Dr. Fink, who developed a method of sequencing discrete sound events for soundscape synthesis that is absent from this document but can be found in his dissertation. Dr. Thornburg was an excellent mentor to whom I have to thank, but not blame, for most of my understanding of digital signal processing and modeling stochastic processes.

Now, the thanks. Many thanks go to all my various friends and mentors with whom

I've had the pleasure of working in the School of Arts, Media, and Engineering. Despite all of its changes in six years, I couldn't have asked to be part of a more delightfully eclectic group. Special thanks go to Drs. Thanassis Rikakis, Hari Sundaram, and Andreas Spanias for their dedication in figuring out "what to do with Brandon" and my entire committee for putting up with my many changes in direction. Dr. Spanias provided much-needed calm guidance to steer me toward completion, and Dr. Perry Cook couldn't have come at a more perfect time to help advise me on matters related to sound texture synthesis. Finally, Dr. Gregory Shrader is to thank for encouraging me to be mindful of all my uses of the word "just" in all its minimizing incarnations.

Most importantly, though, all my love and thanks go to my family and friends (including Nermal, Muffin, Mochi, and Nekobasu) who have, for whatever strange reasons, always entertained my nonsensical, "out-there" ramblings, joined me in speculative jamming, and been there when I most felt like a quaking mess.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

The late 1960's and early '70's, with the formation of the World Soundscape Project at Simon Fraser University, R. Murray Schafer's publication of *The Tuning of the World* [1], and later Barry Truax's *Acoustic Communication* [2], amongst others, saw the beginnings of a new way of looking at the auditory scene known as acoustic (or "soundscape") ecology. An inherently interdisciplinary concept, acoustic ecology seeks to examine the relationships between life and its environmental context as mediated through sound. In the over forty years that have passed since, acoustic ecology has supported the development of numerous new ways of looking at our acoustic environments, inspiring work in fields such as musical composition [3], bioacoustics [4], anthropology [5; 6], and architectural acoustics [7], even proposing that soundscapes be seen as entities worth respecting and preserving in their own right [1; 8]. An ecological interpretation of the acoustic scene suggests a transactional relationship, where auditory phenomena are inseparable from their associated physical sources and fields, and what we perceive to be separate sound objects or events may in fact be intimately interdependent. Similar to other studies of ecology, these interdependent relationships inspire certain prescriptive efforts in increasing awareness, appreciation, and preservation of soundscapes [9; 8].

From a technological perspective, acoustic analysis has also had major breakthroughs. In music, for example, information retrieval and machine learning have opened up such possibilities as query-by-humming (or similarity) [10; 11; 12], automatic music transcription [13], musical source separation [14], genre recognition [15; 16; 17], and other types of semantic annotation [18; 19], to name but a few topics. Similarly, work in speech coding and recognition (e.g. [20; 21]), through developing new strategies for encoding,

compressing, and streaming speech, has assisted the creation of a global wireless telecommunications network, and we are beginning to see the possibilities of speech recognition in consumer applications. Through tailoring many of these techniques toward environmental audio, perhaps we can gain a better understanding of the composition and dynamics of soundscapes.

This document proposes a number of specific algorithms for the analysis of environmental sounds that can make up a suite of computational tools for soundscape studies. Additionally, a significant component of these tools, geographical soundscape synthesis, will focus on assisting experiential studies, where through reflecting on the soundscape as an auditory layer to interactive maps, listeners can intuitively explore its properties geographically.

1.1 The soundscape

Although loosely used to describe the field of sounds representative of a particular place, the term *soundscape* has many interpretations, some of which provide insight into what tools might assist in their study. Within the context of [1], Schafer provides a generic definition, describing a soundscape as any portion of a sonic environment regarded as a field of study: actual environments, musical compositions, tape montages, and so forth. A more specific definition is used in [2], where Truax describes a model that defines the soundscape as distinct from the *sonic environment*. Specifically, while a sonic environment might describe all the acoustic energy present in a particular place, the soundscape refers to how that environment is understood by those living in it. This communication model has a significant psychological component, as the feedback loop between auditory sensing and production is then the subject of analysis. This interpretation is also reflected in the *Handbook for Acoustic Ecology* [22]. An important consequence of this definition of a soundscape is that the term applies not only to open spaces,

but also to those experiences provided by electroacoustic sources, such as loudspeakers or even noise-canceling headphones: one way in which we actively construct soundscapes is by deliberately replacing acoustic environments with other, more private environments. Other uses of the term extend to music (for example, spatial compositions) and some literature in ethnomusicology, where it can be used to describe the musical stylings of a particular region or time period [23].

Despite the interdependent nature of a soundscape, certain classifications of sound events and streams can be useful for description. To start with, Schafer provides the following concepts:

1. Keynote sounds: similar to the key of a musical piece, a keynote sound can be described as the ground, as opposed to figure, of a soundscape. Schafer states that it may not always be consciously audible, but it nevertheless characterizes the rest of the experience of the environment. Traffic, wahwah, wind, and other types of longterm background dins are examples.

2. Sound signals: as the outstanding figure of a soundscape, these foreground sounds are those events to which we tend to consciously attune. Examples include sirens, bells, and other outstanding events.

3. Soundmark: either ground or signal, soundmarks uniquely characterize a place, just as a particular building may characterize a city's skyline or a lone tree may be a landmark to a field.

Schafer also provides a detailed taxonomy of example sounds one might encounter in the field. Krause [2] additionally presents a source-based classification:

1. Geophany: sounds initiated by geophysical phenomena such as climate and geological events.

2. Biophany: sounds initiated by biological life (plants and animals), not including humans.

3. Anthrophony: sounds produced by humans or their artifacts.

1.2 Listening practice

As previously mentioned, any classification scheme, however useful, relies on the notion that a soundscape is comprised of or interpreted as *separate* sound events. Throughout this document, algorithms will often be discussed that treat sonic events an individual segments of single recordings of a soundscape, principally inspired by the concept of auditory streaming, which is supported by a wealth of literature in psychoacoustics, as surveyed by Bregman [24]. Auditory streaming proposes a model of the acoustic scene that is interpreted as several co-occurring streams, which due to cues in sequential (separate in time) grouping and simultaneous (overlapping in time) grouping, can merge and divide, inspired by gestalt grouping principles in visual perception. For example, if two simultaneous tones tend to have similar pitch dynamics, they may be interpreted as a single stream, even if they do not originate from a single physical source: consider, for example, the combined voice of a choir. Similarly, if multiple sound events tend to occur at regular intervals with respect to each other, they may be grouped into a single stream, such as the regular beating of a drum. Additional studies that attempt to examine psychoacoustic phenomenon in an ecological context (i.e. "in the wild") can be found in Neuhoff, et al. [25].

This psychoacoustic perspective has guided many of the decisions made in developing the computational techniques described in later sections. Focusing on the psychological interpretation of the acoustic scene is quite different from any approach that attempts to identify specific "sounds" with individual "sources" in the material world. This focus

also plays a fundamental role in Truax's *Acoustic Communication*. While it may be true that physical interactions transduce acoustic energy, a purely material interpretation does not necessarily capture the experiential phenomenon of audition, possibly misleading whatever analytical tools it informs. Speaking to acoustic ecology, Truax mentions that acoustic communication "attempts to understand the interlocking behavior of sound, the listener, and the environment as a system of relationships, not as isolated entities," and "with sound, everything interacts with everything else." Finally, in his introduction, he mentions that "for any argument based on perceptual experience, the only verification and understanding will come from actual practice." [2] This concept of *listening* as a means to understanding has been the focus of many other composers and theorists.

Gaver [26; 27] mentions two types of listening: *everyday* listening and *musical* listening. Everyday listening focuses on the type of listening that is more common to everyday life. Typically, when we hear sounds, we attempt to associate them with a specific physical source. Musical listening, on the other hand, may best be described in terms of Pierre Schaeffer's treatise on *musique concréte*, *Traité des objets musicaux* [28; 29], in which he describes *acousmatic* compositions comprised of *sound objects*, decontextualized from their physical, historical, or psychological contexts. Schaeffer refers to this style of understanding sound as *reduced* listening. Although he used different terms, the American composer John Cage once alluded to the concepts of reduced listening and sound objects as follows:

> "I love sounds just as they are, and I have no need for them to be anything more than what they are. I don't want them to be psychological. I don't want a sound to pretend that it's a bucket or that it's president or that it's in love with another sound. I just want it to be a sound." [30]

Similar styles of listening have been proposed for the experience of the *soundwalk*, described by Schafer [1], which can be seen as a mindful listening practice while moving

through a space. Also of important note is the use in compositional practice of Pauline Oliveros's *Deep Listening* [31], which presents several compositions, or meditative practices, related to consciousness exploration through sound or one's individual relationship to sound. Many of these styles of understanding and relating to one's environment are inspired by or bear resemblance to mindfulness practice. From the standpoint of Sōtō Zen, rōshi Shunryu Suzuki described an understanding of sound as follows:

> "Sound is everywhere. If you just practice it, there is sound. Do not try to listen to it. If you do not listen to it, the sound is all over. Because you try to hear it, sometimes there is sound, and sometimes there is no sound." [32]

Or, as Alan Watts frequently quoted rōshi Morimoto, "the sound of rain needs no translation." [33] While perhaps these practices cover far more than listening alone, they help to describe the concept of a completely interconnected ecology of sound that is constantly evolving with or without intellectual interpretation.

1.3 Problem statement and contributions

Schafer describes the problem of large-scale analysis of soundscapes as follows:

> "To give a totally convincing image of a soundscape would involve extraordinary skill and patience: thousands of recordings would have to be made, tens of thousand of measurements would have to be taken, and a new means of description would have to be devised." [1]

While this remark was written before many of the advances in digital audio recording and signal processing that we have at our disposal today, the description of a "convincing image of a soundscape" is still relevant, as it lays out many of the necessary components to any soundscape organization project, automated or not. How one collects data, measures it, and describes it defines what is being communicated by any representation, as each

process introduces significant bias. By understanding what these biases are, we can better understand what our findings do and do not tell us.

This dissertation will attempt to address the problem of providing a convincing image of a soundscape by providing tools that answer the following questions:

1. How can we analytically decompose multiple, long field recordings into understandable components?

2. How can we catalog these components and learn from the relationships between auditory features and semantic annotations?

3. How can we provide a new experiential means of reflecting on the soundscape to help explore relationships that might be missed in computational analysis?

These questions are addressed through the following contributions:

1. An open source tool, *Sirens*, for the segmentation and comparison of continuous environmental field recordings, allowing users to automatically decompose long recordings into individual sonic events and then compare how similar they are to each other in terms of acoustic content.

2. A unified graph-based framework that allows for the indexing of sound events and associated semantic tags, making use of both acoustic similarity from *Sirens* and WordNet-derived metrics of semantic similarity to allow for both content-based retrieval (query by example), text-based retrieval, and automatic annotation of sounds. Results are presented that suggest this type of associative retrieval could greatly benefit existing large sound databases, especially in the realm of querying databases with tags that do not yet exist in the database.

3. A review of existing interactive sound mapping projects, evaluating their various features, content, and scope.

4. The evaluation of several new sound texture synthesis techniques for parametrically morphing between multiple source sound textures for use in an interactive sound map.

1.4 Outline

The remaining chapters will describe specific computational tools to assist in the study of soundscapes and environmental sound, falling roughly into two categories: analysis and synthesis. For analysis, Chapter 2 will discuss an open-source environmental sound analysis library, *Sirens*, used for segmenting continuous field recordings into discrete events and evaluating the similarities between them. Chapter 3 will discuss how these events can then be indexed into a large graph-based data structure consisting of both the sound events and folksonomy-based tags, where acoustic similarity and semantic similarity are used to allow both text-based and content-based retrieval and annotation. Moving on to synthesis, Chapter 4 will provide a brief survey of nearly one hundred existing interactive maps of field recordings, discussing their different types of content and interaction styles. Pointing toward a new type of interactive experience of acoustic maps, Chapter 5 will discuss a model for the automatic synthesis of soundscapes sourced from multiple, geotagged field recordings through the parametric mixing of background sound textures. Finally, Chapter 6 will discuss how these tools can be used as a whole and several possible future directions in the realm of computational soundscape studies.

Chapter 2

SIRENS: SEGMENTING AND COMPARING SOUND EVENTS

One of the central issues with analyzing entire soundscapes is sheer volume. Soundscapes cover vast distances and long durations. For example, a stationary field recording, no matter how long, represents a single point in an enormous geographic region, just as a still photograph represents a single point in time. Considering Schafer's remarks regarding providing convincing representations, multimedia retrieval and indexing paradigms which can deal with large numbers of sounds, measurements, and annotations may provide useful ways in which to organize the soundscape. Central to these retrieval schemes is finding suitable representations or features to index. These representations could be explicit, such as individually recorded sound events, or they could take the form of measurements, such as spatial loudness contours or even affective descriptors [34].

For the purpose of recording soundscapes, using long, continuous recordings can greatly reduce the number of man-hours spent editing and collecting data. Additionally, continuous recordings ensure that more activity in a space is recorded, rather than just those events which one targets based on preconceptions about what *should* occur in the space. These preconceptions can be seen as specific hypotheses to be tested, but they become restrictive when performing exploratory analyses or when the potential causes of certain phenomena are unknown. Of course, where a microphone is placed or how it moves also biases what activity is recorded in a space, but with continuous recordings, the process of data collection may not require as frequent of high-level decisions about what must be recorded. Finally, once specific events are identified in a continuous recording, one may recover the original context in which they have occurred simply by listening to as much of the surrounding material as desired [35]. This latter idea of continuous

recordings as an aid to thought and memory was part of the motivation behind Vannevar Bush's MEMEX device proposed in 1945 [36; 37].

To this end, the open source C++ library, *Sirens* allows for the segmentation, indexing, and retrieval of environmental and natural sounds. *Sirens* allows any developer to automatically

1. extract feature trajectories from long recordings,
2. segment these recordings into many short, independent sound events, and
3. compare these recordings to each other in terms of acoustic content.

With these features, *Sirens* provides the backbone for the retrieval framework that will be described in Chapter 3. *Sirens* is freely-available open-source software that is architecturally separate from any retrieval or synthesis platform, making it available for other acoustic indexing applications, such as archiving sound effects or personal recordings. Additionally, *Sirens* is built with interoperability with existing recording and audio analysis software in mind, so portions are also available as a collection of *Vamp* [38] plugins, common to acoustic analysis software such as *Sonic Visualiser* [39] and *Audacity* [40]. *Sirens* also supports the use of feature trajectories provided by *Vamp* plugins for the segmentation and comparison processes, allowing developers to choose which feature sets they wish to use for these algorithms, such as features from the *LibXtract* [41] library. This modularity may make the segmentation and retrieval algorithms within *Sirens* useful for the analysis of other acoustic media, including music and speech.

2.1 Feature extraction

Indexing and retrieval schemes for audio clips typically start with some means of acoustic feature extraction (for some examples, see [42], [43], and [10]) and are commonly known as *content-based retrieval* schemes. Typically, this type of analysis involves

extracting several feature trajectories across the duration of an acoustic signal according to a frame-based approach, whereby features are computed across a sliding window of the original signal, as the characteristics being measured (such as loudness or pitch) are generally non-stationary over these durations. These windows typically vary from 20-100ms and may overlap. For example, a feature calculated on a one-second sound clip with a 20ms window and half-overlap will result in a feature signal of 100 frames. For *Sirens*, these features are computed over frames obtained by convolving the audio signal with a 20ms hamming window with 10ms overlap.

With *Sirens*, we are primarily interested in analyzing entire sound environments, which encompass a broad variety of sounds compared to systems that are intended for specific sources, such as recorded music [44], birdsong [45], or bat vocalizations [46]. As such, we have chosen a multifeatured approach, which has significant advantages over using single features (such as solely relying on mel-frequency cepstral coefficients (MFCCs) [47], commonly used in musical analysis), as environmental sound events tend to vary over a large range of perceivable characteristics. We have developed a small, core set of features that were developed with Bregman's concept of *ecological validity* [24] in mind, focusing on those characteristics which might best distinguish two sound events from each other to a human listener, according to known psychoacoustic phenomena. These features are as follows:

1. Loudness: We define loudness as the RMS level of a windowed frame of data in decibels, that is:

$$L_t := 20 log_{10}(RMS_t), \tag{2.1}$$

   where L_t and RMS_t denote the loudness and RMS level of the t-th windowed frame, respectively.

   Loudness has clear usefulness for indexing environmental sound events, as two ad-

jacent sound events within the same soundscape may frequently vary in loudness. In some cases, sounds may not be loud enough to be perceivable, and in other cases, significantly loud sound events may event mask softer events [48]. Note that the term "loudness" is used in this instance to refer to a simple, perceptually scaled estimate of instantaneous loudness rather than a nuanced measure from a more complex model.

2. Temporal sparsity: Temporal sparsity is defined as the ratio of the L^∞ and L^1 norms (maximum and sum) of loudness values within a one-second sliding window over computed loudness frames. As such, temporal sparsity responds more slowly to sudden events than might loudness, but it computes frames at the same temporal resolution.

$$TS_t := \frac{\max\left(L_{t-N+1}, ..., L_t\right)}{\sum_{k=t-N+1}^{t} L_k}, \tag{2.2}$$

where N is the number of frames over which the sliding window is computed. For a sliding window one second in duration and a typical frame length of 20ms with half overlap, $N = 50$.

By taking a ratio of the maximum loudness value over the sum of all loudness values, temporal sparsity provides an interesting textural, rather than spectral measure. For example, slow typing on a keyboard might have high temporal sparsity, faster typing will have lower temporal sparsity, and a constant buzzing will have yet lower temporal sparsity, as its textural components are of too high of frequency for the feature to pick up, depending on the window size used for analysis.

3. Spectral centroid: Spectral centroid is defined as the centroid of the bark-weighted frequency distribution over a single frame. Values are scaled by their levels in barks,

a perceptually-adapted frequency measure described in [49].

$$SC_t := \frac{\sum_{j=1}^{M} b_j \left(b_j - b_{j-1}\right) |STFT_{t,j}|^2}{\sum_{j=1}^{M} \left(b_j - b_{j-1}\right) |STFT_{t,j}|^2}, \tag{2.3}$$

where $|X_t(j)|$ refers to magnitude of the j-th STFT coefficient of the t-th frame and b_j refers to the bark weighting coefficient of the bin's median frequency.

Spectral centroid can provide a rough measure of pitch for otherwise non-pitched sounds. For some sounds, such as white noise, traffic, or even running water, this measure will have less significance. For other features, sudden changes in spectral centroid can be a useful indicator of sound event onsets.

4. Spectral sparsity: Spectral sparsity (inspired by the GINI index and related sparsity metrics [50]) is the ratio between the L^∞ and L^1 norms of the STFT coefficients for a particular frame:

$$SS_t := \frac{\max\left(|STFT_{t,1}|, ..., |STFT_{t,M}|\right)}{\sum_{k=1}^{M} |STFT_{t,k}|}, \tag{2.4}$$

Spectral sparsity will be extremely high for sources with notable harmonic components and yet higher for pure-tone-based sounds such as those produced by sirens or alarms. Sounds with more complex frequency structure, such as rushing water or gusts of wind, will have much lower spectral sparsity.

5. Transient index: The transient index is defined as the overall magnitude of the MFCC flux between two frames of data, summed over a number of frames along a sliding window as follows:

$$TI_t := \sum_{k=t-(N+2)}^{t} \left\| MFCC_k - MFCC_{k-1} \right\|_2, \tag{2.5}$$

where $MFCC_k$ is the fifteenth-order MFCC vector for frame k and N denotes the number of frames over which the transient index is computed. Similar to temporal sparsity, generally $N = 50$.

High transient index indicates a rapid fluctuation in spectral content, which has been shown to be a highly effective measure for detecting transients in acoustic signals [51]. Sound events can vary in their abundance of transients. For example, running water typically involves high fluctuations in spectral content, whereas a constant pure tone, such as an alarm, will have a much lower transient index.

6. Harmonicity: Harmonicity is a a probabilistic measure of the likelihood that a particular sound is produced from a harmonic source, such as a human voice [52] or musical instrument. Specifically, it is defined as the maximum likelihood that a particular fundamental frequency and its harmonics are responsible for the spectral structure of a sound, obtained using an efficient approximation of Goldstein's algorithm [53] for pitch estimation [51]. For more information, see [35].

   Environmental soundscapes can contain a large number of sounds, and whether or not a particular sound event emanates from a harmonic source can be an important characteristic in this space, indicating, for example, the presence of human voices or many animal vocalizations.

2.2 Segmentation

Segmentation of audio signals is useful for applications such as browsing and annotation [54], phoneme detection [21], and musical note or chord detection [55]. In the case of environmental soundscapes, we aim to extract individual sound events. Of critical importance to the task of event detection, however, is how we define the term "event." To obtain a useful definition of a sonic event, it is necessary to consider the types of listening described in Chapter 1. For the purposes of the environmental soundscapes we wish to analyze in *Sirens*, we concern ourselves with segmenting events using some inspiration from auditory streaming cues.

2.2.1 Segmenting an Individual Feature

For segmenting sound events from long environmental recordings, *Sirens* provides an efficient implementation of a switching state space model developed by Wichern, et al., described in [35] and [19]. This technique models events as having distinct onsets and durations that may overlap, allowing for multiple sound events to be detected at the same time while not explicitly accounting for multiple physical sources in analysis, such as in source separation algorithms. Additionally, the segmentation framework used in *Sirens* uses a multi-feature approach as opposed to single-feature approaches based on tasks such as transient detection [51] or novelty measures [56]. Among those described in Section 2.1, different features are typically responsive to different types of sounds, depending on the environment in which the sound was recorded, and segmentation boundaries for each feature may be delayed, depending upon the resolution of analysis and behavior of the feature (for example, features calculated along a sliding window, such as temporal sparsity and transient index, are delayed in detecting sudden activity.)

While *Sirens* uses a multi-feature method for segmentation, it is modular in that separate features have individual segmentation trajectories which contribute to the overall segmentation mode. By having each feature independently contribute to the overall prediction of segmentation boundaries, not all features need to be responsive to the existence of a sound event for it to be detected. At the core of segmentation in *Sirens* is a switching-state-space prediction model [57], whereby an optimal trajectory of segmentation states is computed at the same resolution as the feature trajectories, tracking several estimations according to a global Markov state transition model.

Starting with each individual feature, a linear prediction model is defined according to the dynamic bayesian network depicted in Figure 2.1, whose state is estimated with a Kalman filter, thereby modeling each feature as an underlying linear process corrupted

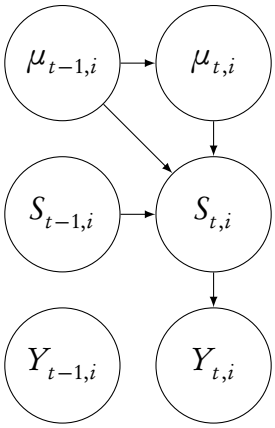


Figure 2.1: Individual feature segmentation model.

by noisy measurement. From this model, we define three variables:

1. $Y_{t,i}$, the observation of feature i at time t. This corresponds to the original feature values extracted in Section 2.1.

2. $S_{t,i}$, the estimated feature value of feature i at time t. Assuming that calculated features are observations of inherent features corrupted by Gaussian noise, this represents the estimated feature trajectory.

3. $\mu_{t,i}$, the estimated segmentation state of feature i at time t, taking on values of the set $\{-, O, C\}$, where $-$ represents the lack of a sound event, O represents the sudden onset of a sound event, and C represents the continuation of a sound event that has already started. How these variables are connected will be shown shortly.

Specifically, the hidden state, S_t, is represented as the following recurrence relation:

$$S_t = (1-\alpha)S_{t-1} + \alpha q_t(\mu_{t-1}, \mu_t), \tag{2.6}$$

$$Y_t = S_t + r_t, \tag{2.7}$$

with q_t and r_t modeled as Gaussian noise as follows:

$$q_t(\mu_{t-1}, \mu_t) \sim \mathcal{N}(0, Q(\mu_{t-1}, \mu_t)) \tag{2.8}$$

$$r_t \sim \mathcal{N}(0, R), \tag{2.9}$$

where R and Q are tunable measurement and process variance, respectively. Note that not all transitions, $\mu_{t-1} \rightarrow \mu_t$ are possible, as will shortly be discussed, significantly reducing the number of free parameters needing to be tuned.

With appropriately tuned covariance for each feature's trajectory, $Y_{t,i}$ and $S_{t,i}$ could be used alone for the purpose of feature noise reduction using a Kalman filter if one already knew the segmentation state. However, the switching state-space model we use assumes that the stochastic dynamics of each feature varies with its segmentation state ($-$, O, or C). At time t, if we track the residual cost of predicting $S_{t,i}$ according to the distribution assumed by state $\mu_{t,i}$, we can choose the state with the least cost to be the actual segmentation state.

For example, loudness will typically encounter a higher value in the presence of a sound event to which it is responsive, but may vary throughout, so the covariance for $-$ and C may differ, that for O likely being higher than either. Similarly, the transient index will likely be quite high at the *onset* of an event, since a sudden change, such as a percussive hit followed by a quieter resonance, might be represented as a transient in the audio signal. We have also found that spectral sparsity has an interesting behavior where the variance of its trajectory is lower during the presence of a sound event. Sound events are typically either very spectrally sparse or not sparse at all (as in the case of harmonic sounds or pure tones), whereas noise to which this feature is not responsive will typically vary in sparsity.

With three states, one could imagine that this estimation will be quite combinatorially expensive, as there exist nine possible transitions per feature, not including their

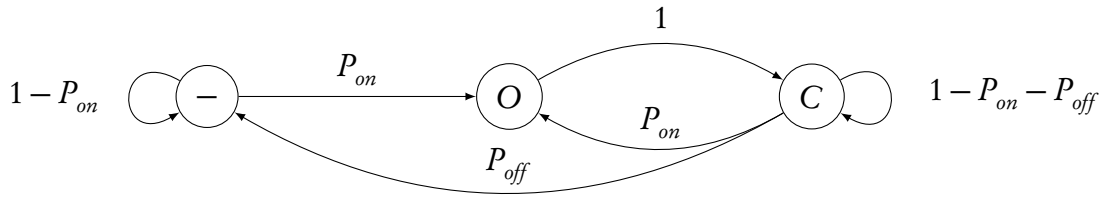


Figure 2.2: Prior Markov transition probabilities for both the per-feature segmentation mode, $\mu_{t,i}$ and the global segmentation mode, M_t. Note that P_{on} and P_{off} form a simple pair of parameters for tuning the global state transitions.

fusion into a global segmentation state. However, not all transitions are possible. For example, a sound event must onset before it can continue to be detected ($\mu_{t,i} = C$ if and only if $\mu_{t-1,i} = O$) and an event will not onset if it is to only turn off the next frame without continuing to be detected ($\mu_{t,i} \neq -$ if $\mu_{t-1,i} = O$). We can therefore reduce the combinatorial complexity by ensuring that only logical state transitions are allowed, according to the Markov state transition diagram shown in Figure 2.2.

In addition to determining which state transitions are acceptable, this transition network sets a prior distribution on which state transitions are more likely, which provides a means of tuning certain features to make them more or less responsive or more or less "sticky" in the case of ambiguities between their trajectory distributions. P_{on} and P_{off} further simplify the parameterization, acting as tuning parameters for the prior probability that the feature's segmentation state will onset or turn off, respectively.

2.2.2 Multi-feature fusion

In addition to each feature's segmentation state, there also exists a global segmentation state, which is the primary focus of our estimation. Since not all features respond to certain events and some respond with more delay than others, we can combine them into a global switching state space model. Figure 2.3 shows how the global segmentation

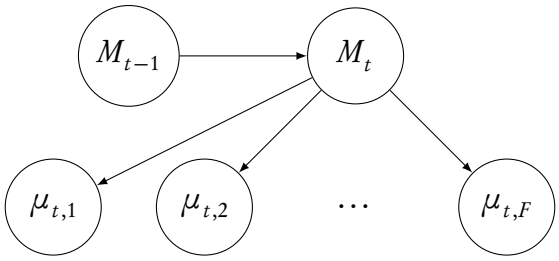


Figure 2.3: Inference of global segmentation state from each feature's segmentation state.

state is conditionally dependent upon each feature's state.

By assuming that the global state transition probabilities behave similarly to those of individual features, we can fuse the prediction of all feature segmentation states and the global segmentation state into one model, which can be used to approximate the state sequence with maximum likelihood via the Viterbi algorithm as per [57]. For full details on the analysis of this model developed by Wichern et al., including performance evaluations and parameter assignments, see [19].

The global mode transition probabilities, $P(\mu_{t,i}|\mu_{t-1,i}, M_{t-1}, M_t)$ are listed in [19] as follows:

M_{t-1}	M_t	$\mu_{t-1,i}$	$P\left(\mu_{t,i}=-\right)$	$P\left(\mu_{t,i}=O\right)$	$P\left(\mu_{t,i}=C\right)$
$-$	$-$	$-$	1	0	0
$\{-,O,C\}$	$-$	$\{O,C\}$	$1-p_{lag+,i}$	0	$p_{lag+,i}$
$\{-,O,C\}$	$\{O,C\}$	$-$	$1-p_{lag+,i}$	$p_{lag+,i}$	0
$\{-,C\}$	O	$\{O,C\}$	$p_{lag+,i}-\left(p_{lag+,i}\,p_{lag+,i}\right)$	$1-p_{lag+,i}$	$p_{lag+,i}\,p_{lag+,i}$
C	C	$\{O,C\}$	0	0	1
O	C	O	0	0	1
O	C	C	$p_{lag+,i}-\left(p_{lag+,i}\,p_{lag+,i}\right)$	$1-p_{lag+,i}$	$p_{lag+,i}\,p_{lag+,i}$

Table 2.1: Transition probabilities for inferring the global state from the individual state variables, $P(\mu_{t,i}|\mu_{t-1,i}, M_{t-1}, M_t)$, where i denotes the feature index for each variable.

In addition to limiting the segmentation transition probabilities as previously mentioned, the variables $p_{lag+,i}$ and $p_{lag+,i}$ allow for certain features to detect event changes as lagging behind or pre-empting the global segmentation mode. Features such as temporal sparsity, which is calculated over a long sliding window, for example, may react quite late, having a higher lag.

To summarize, the model takes the following parameters. First, we have nine parameters per feature, Y_i:

1. $Q_i(\mu_{t-1,i}, \mu_{t,i})$, where $\mu_{t-1,i} = -, \mu_{t,i} = -$, feature variance for lack of an event.
2. $Q_i(\mu_{t-1,i}, \mu_{t,i})$, where $\mu_{t-1,i} = O, \mu_{t,i} = O$, feature variance for a segment staying on.
3. $Q_i(\mu_{t-1,i}, \mu_{t,i})$, where $\mu_{t-1,i} = O, \mu_{t,i} = -$, feature variance for a segment turning off.
4. $Q_i(\mu_{t-1,i}, \mu_{t,i})$, where $\mu_{t-1,i} = -, \mu_{t,i} = O$, feature variance for an onset of a new segment.
5. $Q_i(\mu_{t-1,i}, \mu_{t,i})$, where $\mu_{t-1,i} = O, \mu_{t,i} = O$: feature variance for an onset during an existing segment.
6. $\alpha_i \in [0, 1]$: Low-pass filter coefficient for the feature.
7. r_i: Observation noise for the feature.
8. $p_{lag+,i} \in [0, 1]$: Probability that the feature's segmentation mode will lag a frame behind the global mode.
9. $p_{lag-,i} \in [0, 1]$: Probability of a frame detecting onset early.

For the global mode, we also have global priors P_{on} and P_{off}, so for F features, the model is tuned with $9F+2$ parameters. In the case of the six primary features in *Sirens*, this corresponds to 56 parameters, though often parameter values, such as those for $p_{lag+,i}$ and $p_{lag+,i}$ are similar across multiple features. For the purpose of our experiments, we manually tune the remaining parameters through inspection of the feature trajectories against a segmentation by hand. In the future, it may prove fruitful to provide a simpler parameterization or a more automated way to learn these parameters, such as through expectation maximization [58; 59] or a visual interactive tool to allow labeling of states in training data.

2.3 Sound event similarity

The second purpose of *Sirens* is to provide some means of comparing sound events after they have been segmented. Given a source sound, s_i, and a target sound, s_j, [60] describes a method whereby each feature trajectory for s_i is represented as a hidden Markov model (also see [19; 59]) and the likelihood of s_j being produced by the same stochastic process that produced s_i is computed as the joint emission likelihood of the feature trajectories of s_j. This process can be summarized as follows:

1. First, each feature of s_i is represented as either a constant, linear, or quadratic curve according to that which has the best fit. (Optionally, these feature trajectories are first smoothed with a Savitzky-Golay filter [60].)

2. A hidden Markov model is constructed with states along the estimated curves (see Figure 2.4), represented by $\lambda^{(1:F)}(s_i)$.

3. The likelihood of the feature trajectories for s_j, $Y_{1:T}^{(1:F)}(s_j)$, resulting from the models for s_i is calculated as:

$$L(s_i, s_j) = -\log P\left(Y_{1:T}^{(1:F)}(s_j) | \lambda^{(1:F)}(s_i)\right), \tag{2.10}$$

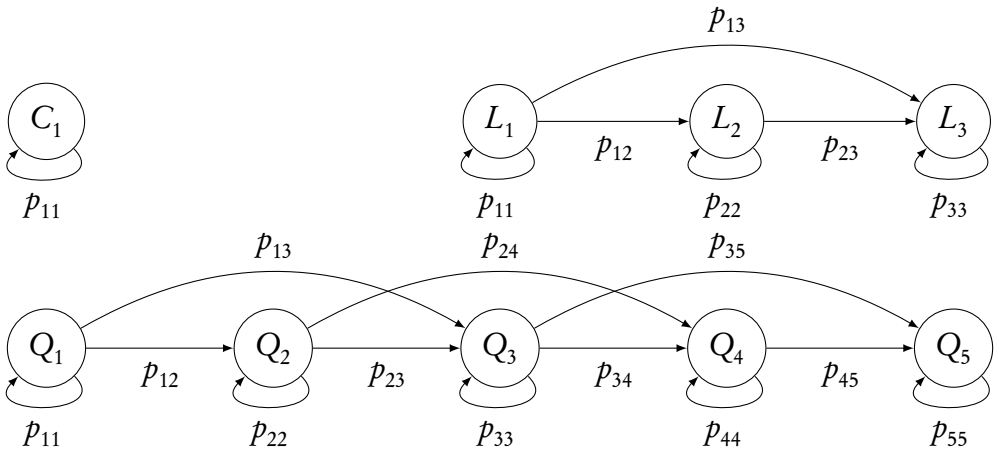


Figure 2.4: Markov transition diagrams, λ^i, for constant (C), linear (L), and quadratic (Q) trajectories.

which can be computed using the forward algorithm.

One benefit of this model over other methods of computing similarity, such as by vector distance between feature trajectories or summary statistics, is that it is robust against varying delays in the feature trajectories, similar to dynamic time warping [61]. Additionally, if we assume the sound events being compared were extracted given the segmentation model from Section 2.2, then this model uses similar representations of the events that were used to identify them. The selection of the possible curve fits is based off the assumption that the sound events are relatively short. For longer events with more complex feature dynamics, HMMs constructed from other representations, such as spline fits, may be more appropriate.

Chapter 3

GRAPH-BASED ACOUSTIC AND SEMANTIC RETRIEVAL

Once sound events have been segmented, it is necessary to index them in terms of relevant features for tasks including annotation, retrieval, and re-synthesis. With this in mind, we have developed a flexible retrieval system that allows for all these retrieval tasks to be performed within a single framework.

Many techniques for text-based retrieval or classification of audio signals are parametric in nature, relying on explicit generalization, where individual classifiers are created for each label. For example, classification systems have been built for automatic record reviews [62], onomatopoetic labels [63], and genre [16], emotion [64], and instrumentation [65; 66] identification. These systems make use of techniques such as one-versus-all discrimination [62], training each label with a support vector machine (SVM) classifier [63; 67], and learning a separate gaussian mixture model (GMM) for each label [18; 62].

These multiclass methods benefit from constant query time complexity independent of the number of training instances, in that it is only necessary for each query (such as a sound, in the case of annotation) to be measured against each classifier. For specific, relatively stationary label domains, such as musical genres, this can be quite beneficial—especially when the number of sounds greatly exceeds the number of labels. However, there are many cases where non-parametric models can provide additional flexibility and robustness. One such case involves the presence of multiple types of information beyond acoustic feature vectors and annotations. For example, [68] and [69] describe methods where *semantic* similarity between *tags* can assist in retrieval and annotation using tags not yet seen in training data. In large-scale systems with more complete tag sets, this may

be less of a problem, but in live databases with incomplete tagging where no large-scale training database exists beyond user activity, as in the case of Freesound [70], retrieval results can often come up empty.

Non-parametric (also known as similarity- or instance-based [71; 72]) schemes compare each query to instances in a live database rather than having distinct training and production / evaluation stages. For example, [73], [74], and [75] use K-nearest-neighbors retrieval, where unlabeled sounds are annotated with tags belonging to their nearest neighbors in an acoustic feature space. [76] and [73] build two separate hierarchical cluster models—one for retrieval and one for annotation.

Non-parametric graph-based techniques are often used for search in semantic and other associative networks. One technique that has seen much use is spreading activation. In spreading activation, an initial node (a query) is labeled with some real-valued, positive weight, and this weight is spread to neighboring nodes, reduced by a decay factor. Spreading activation has been used in textual information retrieval applications, where nodes correspond to documents and terms [77]. Shortest paths are also of interest in associative retrieval. In [78], we introduced a graph-based framework where sounds are connected to tags through user activity and sounds are fully connected via acoustic similarity estimated by the HMM-based query-by-example algorithm from *Sirens* described in Chapter 2. New queries are immediately connected to other sounds or tags (either through acoustic or semantic similarity via the WordNet::Similarity library [79]), and shortest path distances using all nodes are used to rank retrieval results.

The results of [68], which will be discussed in Section 3.3, demonstrate that including semantic similarity to account for tags foreign to the training set can assist in annotation and text-based retrieval both for a random subset from the Freesound library, where tags are associated to sounds in a binary manner, and a smaller comprehensive user study, where each sound is tagged by multiple users. Further, as will be discussed in Section 3.4,

we built upon this graph-based technique and performed a more in-depth study of its properties. Namely, we sought to:

1. evaluate the system using both traditional cross-validation (Section 3.3) and simulations of real-world systems with complete sound and tag sets but incomplete tagging data (Section 3.4),

2. explore variations on using shortest paths for retrieval (Sections 3.2.2 and 3.4.3),

3. demonstrate the effectiveness of adding a shortest-path algorithm to any existing tag-based query system, regardless of the presence of acoustic and semantic similarity measures (Section 3.4.3),

4. improve shortest-path retrieval performance by pruning network edges to nodes' K nearest neighbors (Sections 3.2.3 and 3.4.4), and

5. explore the impact of assigning different weights to the importance of acoustic, semantic, and user-provided information (Sections 3.2.4 and 3.4.5).

3.1 Graph structure

Formally, the network structure for retrieval and annotation takes the form of a weighted, undirected graph, $G = (V, E)$, where $V = S \cup T$ and S and T represent sets of sound and tag nodes, respectively. The graph edges, E, can be partitioned into three disjoint subsets, $E_{SS} \subseteq S \times S$, $E_{ST} \subseteq S \times T$, and $E_{TT} \subseteq T \times T$, representing acoustic, user-provided, and semantic information. The weighting function is denoted by $w : E \rightarrow \mathbb{R}^+$. This type of network structure can be adapted to different domains, such as music or even text documents, but for the sake of this application, we focus on the task of retrieving and annotating environmental sounds. We therefore assume that sounds take the

form of short audio clips representing individual sonic events. The following sections will discuss how the weights for E_{SS}, E_{ST}, and E_{TT} are calculated.

3.1.1 Sound-to-sound weights (E_{SS})

Sound-to-sound weights can be computed by comparing the acoustic content of each sound. From Section 2.3, we obtain the log-likelihood of the feature trajectories of sound s_j being generated from the hidden Markov models, $\lambda^{(1:F)}(s_i)$, expressed as

$$L(s_i, s_j) = -\log P\left(Y_{1:T}^{(1:F)}(s_j) | \lambda^{(1:F)}(s_i)\right). \tag{3.1}$$

For retrieval in the undirected graph, however, it is helpful to have a semi-metric between sounds that is symmetric and nonnegative. In [80], a semi-metric that holds these properties is given:

$$w(s_i, s_j) := L(s_i, s_i) + L(s_j, s_j) - L(s_i, s_j) - L(s_j, s_i). \tag{3.2}$$

3.1.2 Sound-to-tag weights (E_{ST})

Letting $U_{|S|\times|T|}$ be a votes matrix where U_{ij} is equal to the number of users who have tagged sound s_i with tag t_j, we can compute the joint probability of s_i and t_j as

$$P(s_i, t_j) = \frac{U_{ij}}{\sum_{k,l} U_{kl}} \tag{3.3}$$

and set the sound-to-tag weights with a similar logarithmic scaling:

$$w(s_i, t_j) := -\log P(s_i, t_j). \tag{3.4}$$

3.1.3 Tag-to-tag weights (E_{TT})

For tags, we obtain a fully connected tag-to-tag similarity matrix using a similarity metric from the *WordNet::Similarity* library [79]. Specifically, we have tested the `jcn`,

`lin`, `lch`, `res`, and `vector` metrics. The `vector` metric computes the co-occurence of two tags within the collections of words used to describe other tags (their *glosses*) [79] and `lch` is based on the shortest-path distance between two tags in a taxonomy, while `jcn`, `lin`, and `res` are all based on the information content of the terms and their ancestors within a corpus (for a full review, see [81] and [79].)

For consistency in comparing different metrics, we compute the network distances from the following normalization of similarities, $WNS(c_i, c_j)$:

$$w(t_i, t_j) := -\log\left[\frac{WNS(c_i, c_j)}{\max_{i,j} WNS(c_i, c_j)}\right]. \tag{3.5}$$

`jcn`, developed by Jiang and Conrath [82], has an interesting symmetry with the log-scaled weights for E_{SS} and E_{ST} and in some ways can be seen as joining the graph structure with the WordNet taxonomy, as it is based on path lengths between terms weighted by information content. Specifically, for parts of speech that form a hierarchy (e.g. nouns and verbs), you can model the distance between a term and its parent as proportional to the conditional probability of a term, t co-occurring in a corpus also containing its parent, i.e. $P(t|parent(t))$. So, the `jcn` distance between a term and its parent is as follows:

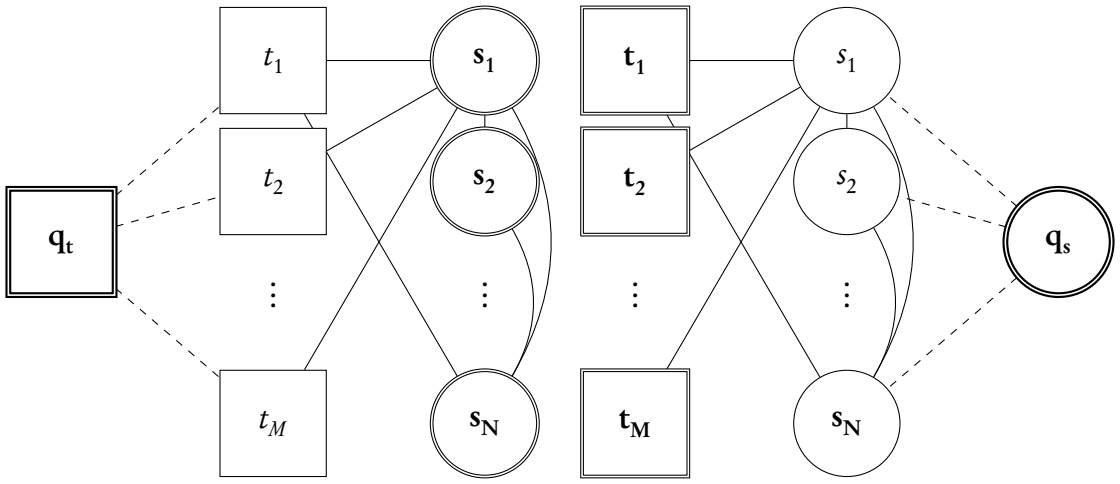
$$\texttt{jcn}(t, parent(t)) = -\log P(t|parent(t)) \tag{3.6}$$

$$= IC(t) - IC(parent(t)), \tag{3.7}$$

where $IC(t)$ denotes the information content of t in a given corpus. Denoting $lso(t_a, t_b)$ as the nearest common parent of two terms, t_a and t_b, (the lowest super-ordinate or most specific common subsumer), one can then compute the distance between any two terms as a weighted shortest path distance between them in the taxonomy [81]:

$$\texttt{jcn}(t_a, t_b) = IC(t_a) + IC(t_b) - 2IC(lso(t_a, t_b)) \tag{3.8}$$

$$= 2\log P(lso(t_a, t_b)) - (\log P(t_a) + \log P(t_b)). \tag{3.9}$$



(a) Text-based retrieval (b) Annotation

Figure 3.1: Two different possible query tasks with a single retrieval network. q_s and q_t represent a sound query and a tag query, respectively, and $t_1, t_2, ..., t_M \in T$ and $s_1, s_2, ..., s_N \in S$ represent tag and sound nodes already in the database. The query node and the subset of nodes over which the user is querying is marked in bold, and on-demand edge weights between the query and its respective class of nodes (sounds or tags) are marked with dashed lines. Note that S forms a clique.

3.2 Path-based retrieval

3.2.1 Shortest path retrieval

Given the structure of the graph, $G(V,E)$ and its weights, w, as defined in the previous section, we rank search results according to their shortest path lengths from the query, q, to the target, t, in ascending order:

$$w^*(q,t) = \min_{P=\langle q,...,t\rangle} \sum_{i=1}^{|P|-1} w(P_i, P_{i+1}), \tag{3.10}$$

which can be computed using Dijkstra's algorithm. Figure 3.1 shows two examples of network queries, one for text-based retrieval, which starts by connecting a foreign tag to

the existing tags and finally retrieving over the sound tags, and annotation, which starts out by comparing a new sound to existing sounds, querying over the entire set of tags.

3.2.2 Depth-ordered retrieval

Shortest paths may sometimes hinder retrieval results in cases where they provide discursive paths that rely on numerous relations. For example, if sound-to-tag weights are trained with ground truth data obtained from user studies or extensive user activity, it would be desirable to only use these direct paths and visit no other nodes rather than second-guessing users (who, for the purpose of evaluation, the literature often assumes are experts).

In these cases, we can form a tuple, L, of positive integers representing desired path depths, with an optional final element, $*$, representing shortest paths of any depth. Any targets unconnected to the query will be returned at the end of the list in random order. For example, $L = (2,*)$ will prioritize minimum-cost direct edges between the query and targets first, only using shortest paths as a last resort in the absence of direct edges. $L = (2,3,*)$ will first return all direct edges, then shortest paths visiting only three nodes, and finally all remaining shortest paths. For the case of $L = (2,3,...,N)$, where depths are in monotonically increasing order, this algorithm works similarly to a breadth-first search. In Section 3.4.3, we will discuss the relative performance of depth orderings.

3.2.3 Edge pruning

Shortest-path retrieval can also be quite computationally expensive. In the worst case, Dijkstra's algorithm has $O(|E|+|V|\log|V|)$ time complexity. As our graph is quite well-connected (for large numbers of sounds), we can assume the complexity of performing a single query is $O(|V|^2)$, as $|E| = O(|V|^2)$ when the number of sounds greatly exceeds the number of tags. In these cases, it can be beneficial to limit search to only a node's K

nearest neighbors, giving a complexity of $O(K|V|+|V|\log|V|) = O(|V|\log|V|)$. Using spectral clustering to cluster sounds (as in [78] and thereby partition E_{SS}, we can even improve retrieval to $O(\log|V|)$ complexity. In Section 3.4.4, we will discuss the effects of K nearest neighbor pruning, where G is converted to a directed graph with inbound and outbound edges of E_{SS} and E_{TT} identical to the original undirected edges and all but the K lowest-weight outbound edges are removed.

3.2.4 Weighting edge classes

Lastly, it should be noted that the ranking of search results can be quite sensitive to variations in weighting between the different classes of edges, E_{SS}, E_{ST}, and E_{TT}, as each assumes a different probabilistic model. If one class has particularly low weights, its edges may be traversed more frequently than edges of other classes. In Section 3.4.5, we examine the effects of setting class-specific weights, γ_C:

$$w_\gamma(n_1,n_2) = \gamma_C w(n_1,n_2)$$
$$\forall(n_1,n_2)\in E_C, \forall E_C \in \{E_{SS},E_{ST},E_{TT}\} \tag{3.11}$$

3.3 Initial evaluation

3.3.1 Methodology

In the evaluation process, two datasets were used. The first, referred to as the *Soundwalks* dataset, contains 178 sound files recorded by the authors. The 178 sound files were recorded during seven separate field recording sessions, lasting anywhere from 10 to 30 minutes each and sampled at 44.1 KHz. Each session was recorded continuously and then hand-segmented by the authors into segments lasting between 2 and 60 seconds in duration. The recordings took place at three light rail stops (75 segments), outside a stadium during a football game (60 segments), at a skatepark (16 segments), and at a college campus

(27 segments). To obtain tags, study participants were directed to a website containing ten random sounds from the set and were asked to provide one or more single-word descriptive tags for each sound. With 90 responses, each sound was tagged an average of 4.62 times. In this initial evaluation, we have used 88 of the most popular tags as our vocabulary.

Because the *Soundwalks* dataset contains multiple subject responses per sound, a real-valued votes matrix can be used to determine the sound-to-tag link weights. Obtaining this votes matrix requires large amounts of subject time, thus limiting its size. So, to test the retrieval network performance on a larger dataset, we used 2064 sound files and a 377-tag vocabulary from *Freesound.org* [70]. In the *Freesound* dataset, tags were applied in a binary (relevant/irrelevant) manner to each sound file by users of the website. The sound files were randomly selected from among all files on the site that contain any of the 50 most used tags and are between 3 and 60 seconds in length. Additionally, each sound file contains between three and eight tags, and each of the 377 tags in the vocabulary were used to describe at least five sound files.

To evaluate the performance of the retrieval network, we adopted a two-fold cross-validation approach wherein each dataset was randomly partitioned into two non-overlapping subsets. One of these subsets and its associated tags was then used to build the retrieval network. The remaining subset was then used as queries to test both the annotation and text-based retrieval performance for unlabeled environmental sounds. Furthermore, an important novelty in this work is the ability of the retrieval network to handle *out-of-vocabularly* tags, that is, querying with tags that are not yet tagged to any sounds. To test the performance for out-of-vocabulary tags, a second tier of cross-validation was employed where all *tags* in the vocabulary were randomly partitioned into five non-overlapping subsets. One of these subsets was then used along with the subset of sound files to build the retrieval network, while the remaining tags were held out of the vocab-

	Soundwalks	*Freesound*
Number of sound files	**178**	**2064**
In network (training)	89	1032
Out of network (testing)	89	1032
Number of tags	**88**	**377**
In vocabulary	18	75
Out of vocabulary	70	302

Table 3.1: Database partitioning procedure for each cross-validation run.

ularly stored in the network. This partitioning procedure is summarized in Table 3.1 for both the *Soundwalks* and *Freesound* datasets. Reported results are the average over these ten (five tag partitonings and two sound partitioning) cross-validation runs. Relevance is determined to be nonzero if a withheld sound file was actually labeled by a user with a tag.

For each performance task, in addition to displaying precision and recall curves, we compute *mean average precision* (MAP), the mean of precision values at the points where each relevant item is returned, and *mean area under the receiver operator characteristic* (MAROC), the area under the curve produced by plotting the ratio of true positives versus false positives.

3.3.2 Semenatic similarity metrics

Table 3.2 lists MAP and MAROC values for both tasks using a variety of different semantic similarity measures. The Jiang and Conrath measure [82] seems to perform best, which is consistent with findings for other applications, such as determining word senses in text [83]. Precision values are much lower for both tasks than corresponding tasks

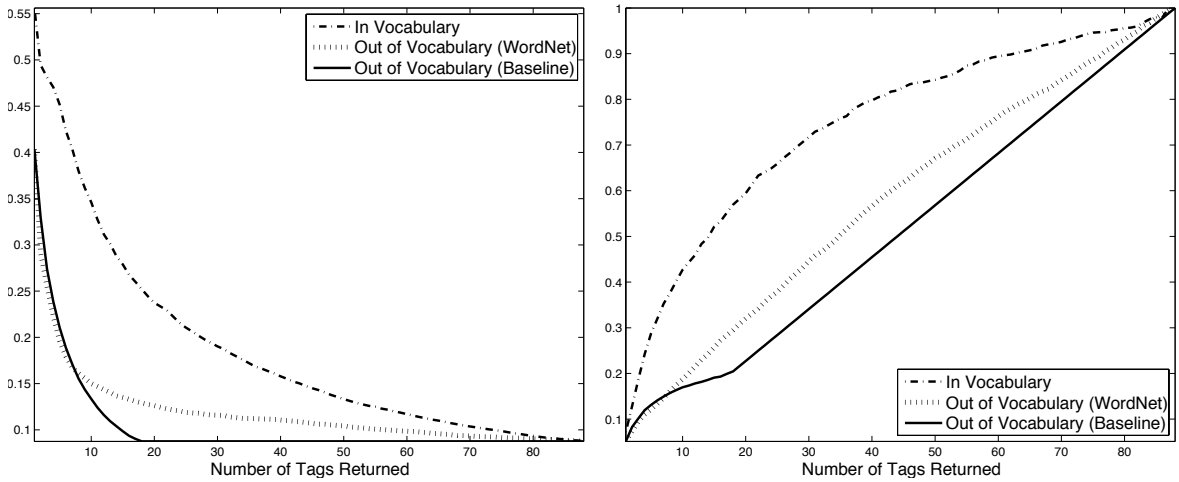
	text-based retrieval		annotation	
Measure	MAP	MAROC	MAP	MAROC
`jcn`	**0.2035**	0.6912	**0.7947**	**0.9045**
`lin`	0.1801	0.7211	0.5669	0.8067
`lch`	0.1831	0.6823	0.5560	0.7890
`res`	0.1941	**0.7212**	0.6023	0.8371
`vector`	0.1720	0.6855	0.5987	0.8305

Table 3.2: MAP and MAROC performance metrics for text-based retrieval and annotation using different WordNet similarity metrics. The highest values are in bold.

in [78; 18; 12], however, it should be noted that the use of foreign terms makes the tasks inherently more difficult. MAROC is quite high in all cases, but this could be misleading, as using the top 88 most popular tags may inflate recall significantly. However, from this data we are able to form a clear idea of which semantic similarity measures work best for audio annotation and retrieval. Since `jcn` seemed to perform best, the remainder of results reported used this metric for setting network weights.

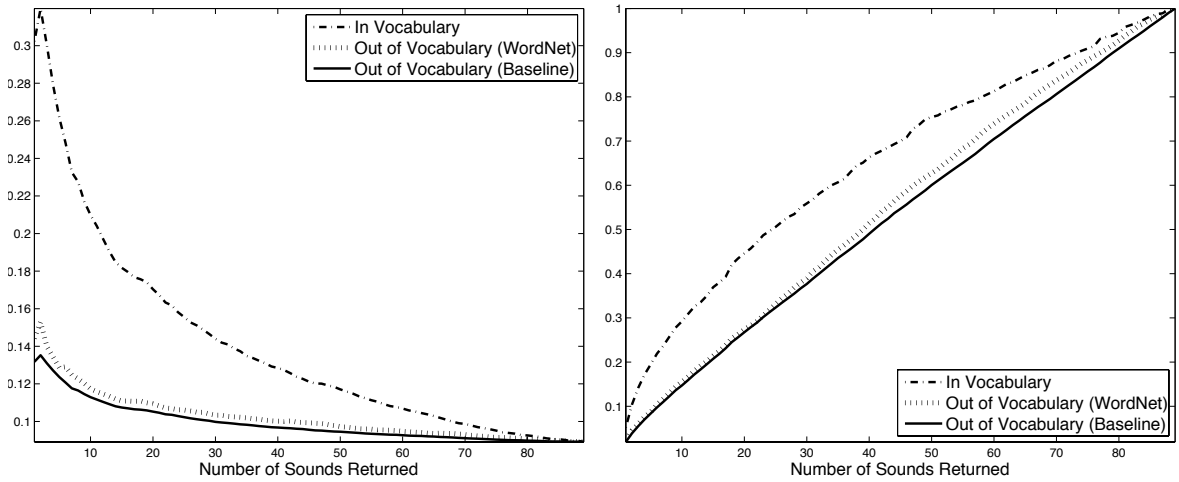
3.3.3 Annotation

Each query returns an ordered list of nodes (tags for annotation and sounds for retrieval), sorted by cumulative path weight in ascending order. An item in this list is said to be *relevant* if it is connected to the query at least once in the original user tagging data. Using this list of relevance for each item returned, we can compute mean *precision*, the percentage of items returned that are relevant as more items are returned, and mean *recall*, the percentage of all relevant items that have been returned. Plotting precision as a function of recall is a useful way of comparing different schemes. Additionally, one can compute performance metrics including MAP and MAROC values. Figures 3.2a and 3.2b



(a) Precision (b) Recall

Figure 3.2: Precision and recall curves for annotation of unlabeled sound files in the *Soundwalks* dataset averaged over 10 cross-validation trials.



(a) Precision (b) Recall

Figure 3.3: Precision and recall curves for text-based retrieval of unlabeled sound files in the *Soundwalks* dataset averaged over 10 cross-validation trials.

display the precision and recall curves, respectively, averaged over all sound queries and cross-validation runs for the *Soundwalks* dataset. The three curves in Figure 3.2 represent

	Soundwalks		*Freesound*	
	MAP	MAROC	MAP	MAROC
In-vocabulary (upper bound)	0.4333	0.7523	0.4113	0.8422
Out-of-vocabulary (WordNet)	**0.2131**	**0.6322**	**0.1123**	**0.6279**
Out-of-vocabulary (Baseline)	0.1789	0.5353	0.1092	0.5387

Table 3.3: Annotation performance using out-of-vocabulary semantic tags. Highest MAP and MAROC scores (out of those not including the upper bound) are in bold.

three different ways of building the retrieval network. The *in-vocabulary* curve can be considered as an upper-bound of annotation performance as all tags are used in building the network. The *out-of-vocabulary (WordNet)* curve uses only a subset of tags to build the retrieval network, and the remaining tags are connected only through tag-to-tag links. The *out-of-vocabularly (Baseline)* curve uses only a subset of tags to build the retrieval network, and the remaining tags are returned in random order. This is a standard parametric approach of training a classifier for each tag would behave for *out-of-vocabulary* tags, as it would have no better heuristic for returning tag suggestions.

From Figures 3.2a and 3.2b, we can see that out-of-vocabulary performance is improved both in terms of precision and recall when WordNet link weights are included. Additionally, from the precision curve of Figure 3.2a, we see that approximately 15% of the top 20 out-of-vocabulary tags are relevant, while for in-vocabulary tags, this number is 25%. Considering the difficulty of the out-of-vocabulary problem and that each sound file is labeled with much fewer than 20 tags, this performance is quite promising. From the recall curve of Figure 3.2b, approximately 30% of relevant out-of-vocabulary tags are returned in the top 20, compared to approximately 60% of in-vocabulary tags. Table 3.3 contains the MAP and MAROC values for both the *Soundwalks* and *Freesound* datasets. We see that performance is comparable between the two datasets, despite the

	Soundwalks		*Freesound*	
	MAP	MAROC	MAP	MAROC
In-vocabulary (upper bound)	0.2725	0.6846	0.2198	0.7100
Out-of-vocabulary (WordNet)	**0.1707**	**0.6291**	**0.0681**	**0.5788**
Out-of-vocabulary (Baseline)	0.1283	0.5355	0.0547	0.5414

Table 3.4: Text-based retrieval performance using out-of-vocabulary tags. Highest MAP and MAROC scores (out of those not including the upper bound) are in bold.

Freesound set being an order of magnitude larger. The slightly better performance on the *Soundwalks* dataset is most likely due to the large amount of social information contained in the real-valued votes matrix that is used to set sound-to-tag link weights. The in-vocabulary MAP values of 0.4333 and 0.4113 compare favorably to the per-word MAP value of 0.179 reported in [18] for annotating BBC sound effects. Benchmarking performance for out-of-vocabulary tags is more difficult, however, as this task has not been considered in the literature.

3.3.4 Text-based retrieval

In text-based retrieval, each tag is used as a query to provide an output distribution over the test sounds. For a given query, we denote by $A(q_c)$ the set of relevant test sounds that are labeled with the query word, and $|A|$ as the number of relevant test sounds for that query. Precision, recall, MAP, and MAROC values are then computed as described previously. Figures 3.3a and 3.3b display the precision and recall curves, respectively, averaged over all sound queries and cross-validation runs for the *Soundwalks* dataset, while Table 3.4 displays the MAP and MAROC values. As with annotation, text-based retrieval with out-of-vocabulary tags is significantly more difficult than with in-vocabulary tags, but including the tag-to-tag links based on the measure of WordNet similarity appears to

Score	Nodes in shortest path				Relevant?
0.19	rail	→ train	→ 094.wav (*train bell*)	→ 165.wav (*traffic/train horn*)	No
0.17	rail	→ voice	→ 136.wav (*pa voice*)	→ 133.wav (*pa voice*)	Yes
0.15	rail	→ train	→ 040.wav (*train brakes*)	→ 030.wav (*train bell/brakes*)	Yes
0.09	rail	→ train	→ 040.wav (*train brakes*)	→ 147.wav (*train horn*)	Yes

Table 3.5: Top four results from *Soundwalks* dataset for text-based retrieval with out-of-vocabulary query q_c = "rail." Parenthetical descriptions are not actual tags, but are provided to give an idea of the content of the sound files. "pa voice" refers to a public announcement of an approaching train.

strengthen retrieval performance.

As with any evaluation against user-provided data, however, what we assume to be ground truth tagging data may in fact be quite variable without a significantly larger dataset. To demonstrate that retrieval performance may be considerably better than the reported precision, recall, MAP, and MAROC performance averaged over noisy tags contributed by non-expert users, we provide the example of Table 3.5. Table 3.5 displays the posterior probability of each of the top four results, the shortest path of nodes from the query to the output sounds, and whether or not the output sound is relevant. The top result is the sound mixture of automobile traffic and a train horn but is not tagged by any users with words similar to "rail," even though, similar to the sounds actually tagged with "rail," it is a recording of a train station. Although filtering these types of results would improve quantitative performance, it would require listening to thousands of sound files and overruling subjective decisions made by the users who listened to and labeled the sounds. Having isolated these sound events from their soundscape contexts, the train horn in 165.*wav* may not evoke the concept of a train for many listeners.

	Soundwalks		*Freesound*	
	MAP	MAROC	MAP	MAROC
With WordNet	0.2166	0.6133	0.2983	0.6670
Without WorNet	**0.3744**	**0.6656**	**0.4633**	**0.7978**

Table 3.6: Performance of retrieval tasks with the *Soundwalks* dataset using WordNet connections between in-vocabulary tags. The best-scoring values are in bold.

3.3.5 In-vocabulary semantic information

Effective annotation and retrieval for out-of-vocabulary tags requires some method of relating the semantic similarity of tags, for example, the WordNet similarity metric used in this work. In this section, we examine how the inclusion of semantic connections between in-vocabulary tags affects annotation and text-based retrieval performance. Table 3.6 compares the MAP and MAROC values for the *Soundwalks* dataset where all tags are used in building the network both with and without semantic links connecting tags. The result of Table 3.6 suggests that when the information connecting sounds and tags is available (i.e. tags are in the vocabulary), the semantic links provided by WordNet confound the system by allowing for possibly irrelevant relationships between tags. This is not unlike the observations of [84] where using WordNet did not significantly improve information retrieval performance. Therefore, future studies have only used semantic links to connect out-of-vocabulary tag queries to other tags in text-based retrieval.

3.4 Evaluation of graph and retrieval variants

3.4.1 Training data

In studies used to evaluate the variations of graph structure (edge pruning, varying edge weights) and retrieval method (depth-ordered retrieval), reported in [85], the same

dataset as in the previous section is used. However, in these studies, we extended the set of tags used from the *Soundwalks* dataset to include all 612 subject-provided tags to more accurately study the system's performance. The *Freesound* dataset remains the same.

3.4.2 Evaluation methodology

For multi-class retrieval, where classifiers are trained for each search term, evaluation procedures typically involve cross-validation, where the set of sounds and their associated tags are split into several (e.g. 10) random non-overlapping subsets, the classifiers are trained with only one subset, and the remaining sounds are used as queries to test the performance of the trained classifiers. With a sufficiently large training dataset, performance results should converge to give a picture of the expected performance in a production setting.

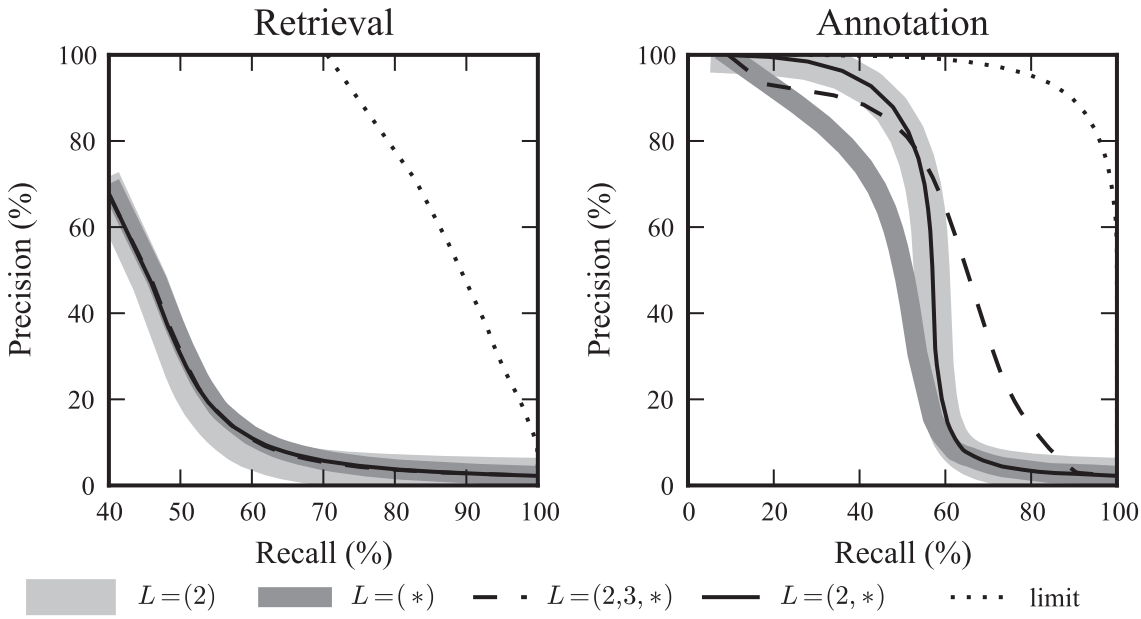
In the studies of Section 3.3 (as well as in [68], [78], and [69]), this evaluation technique was employed for shortest-path retrieval. For the cases of retrieval and annotation using sounds and tags not present in the training data (thereby testing the usefulness of both acoustic and semantic similarity), sounds and tags were split into 2 and 5 subsets, respectively, each combination thereof (one of $2 \times 5 = 10$) being used to build the network. For annotation, out-of-network sound queries were independently introduced to the network by computing their similarity to all other sounds in the network, and out-of-network tags were connected only to the in-network tags. Query performance was tested by querying each out-of-network sound using out-of-network tags. For retrieval, tag queries were connected independently to other tags, and out-of-network sounds were connected only to in-network sounds.

However, this method of cross-validation may not be entirely appropriate for shortest-path retrieval or other associative retrieval algorithms, as there is no distinct training phase. Rather than having only sounds and tags as training data, acoustic and semantic

similarities *between* training instances must be considered–that is, not only the nodes, but also the edges of the graph make up the training data. For this reason, in [85], we chose to implement a different evaluation strategy to compare techniques. In this strategy, we simulate a database where the set of sounds and tags are complete (there are no cross-validation splits), but only a random subset of the user tagging data is available. Specifically, for each association between a sound and a tag (for which there may be many for a single sound-tag pair in the *Soundwalks* dataset), we remove it with 50% probability. For annotation, every sound is used to query the entire set of tags, and for retrieval, every tag is used to query the entire set of sounds. Relevance results are then averaged over each query and over 50-100 trials with different random reductions of the tagging data. This simulation is perhaps more appropriate than the networks built for cross-validation in Section 3.3, as we can examine how using shortest-path retrieval can help make up for sparse tagging data, which is oftentimes present in online tagging systems.

3.4.3 Depth-ordered retrieval

In Figures 3.4 and 3.5, we examine the effects of a) using shortest paths versus retrieving items based only on their tags (as most tag-based search strategies do) and b) using different depth ordering strategies. $L = (2)$ corresponds to the case where only direct tag-to-tag links are used for retrieval (the baseline case, given no acoustic similarity information), $L = (*)$ represents the case of ranking results based on the lengths of their shortest paths, and $L = (2, *)$ and $L = (2, 3, *)$ represent the cases of returning minimum 2- and 3-node paths before resorting to shortest paths. Results are shown for both the *Freesound* and *Soundwalks* datasets. *limit* represents the theoretical upper limit on performance imposed by the dataset, where the ground truth user tagging data itself is used to order results, analogous to *UpperBnd* from [18]. Acoustic links were included for the *Soundwalks* dataset but not the *Freesound* dataset, in order to study the effects of using

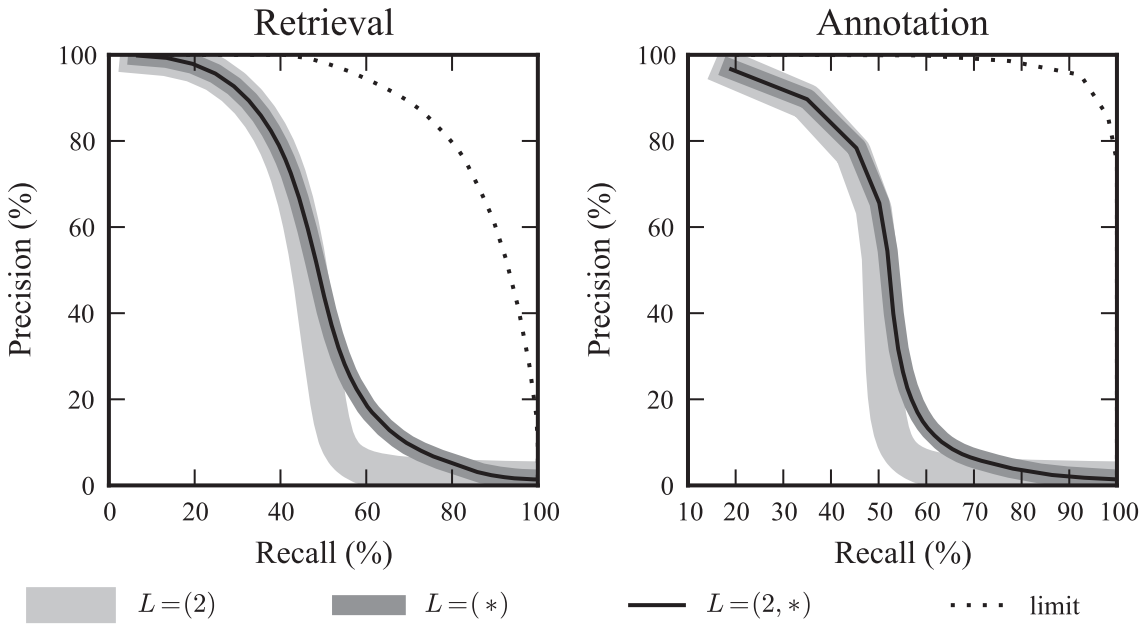


(a) Soundwalks Precision/Recall

	Retrieval		Annotation	
L	MAP	MAROC	MAP	MAROC
(2)	0.5505	0.7615	0.5920	0.7814
$(*)$	**0.5751**	**0.7910**	0.5230	0.7833
$(2,*)$	0.5673	0.7887	0.5964	0.7919
$(2,3,*)$	0.5678	0.7849	**0.6590**	**0.8824**

(b) Soundwalks MAP/MAROC

Figure 3.4: Performance metrics for text-based retrieval and annotation of sounds, respectively, for the *Soundwalks* dataset. Data is averaged across $n = 50$ trials with half the tagging data missing. Curves are labeled according to the order of path lengths used in sorting results, where $*$ denotes all shortest paths. *limit* is the absolute best performance possible with the dataset.



(a) Freesound Precision/Recall

	Retrieval		Annotation	
L	MAP	MAROC	MAP	MAROC
(2)	0.5200	0.7504	0.5237	0.7506
$(*)$	0.5594	0.8610	**0.5628**	**0.8403**
$(2, *)$	**0.5618**	**0.8623**	0.5618	0.8399

(b) Freesound MAP/MAROC

Figure 3.5: Performance metrics for text-based retrieval and annotation of sounds, respectively, for the *Freesound* dataset. Data is averaged across $n = 50$ trials with half the tagging data missing. Curves are labeled according to the order of path lengths used in sorting results, where $*$ denotes all shortest paths. *limit* is the absolute best performance possible with the dataset.

shortest-path retrieval as a drop-in method in an existing system, which at the time of this writing has no acoustic similarity measure.

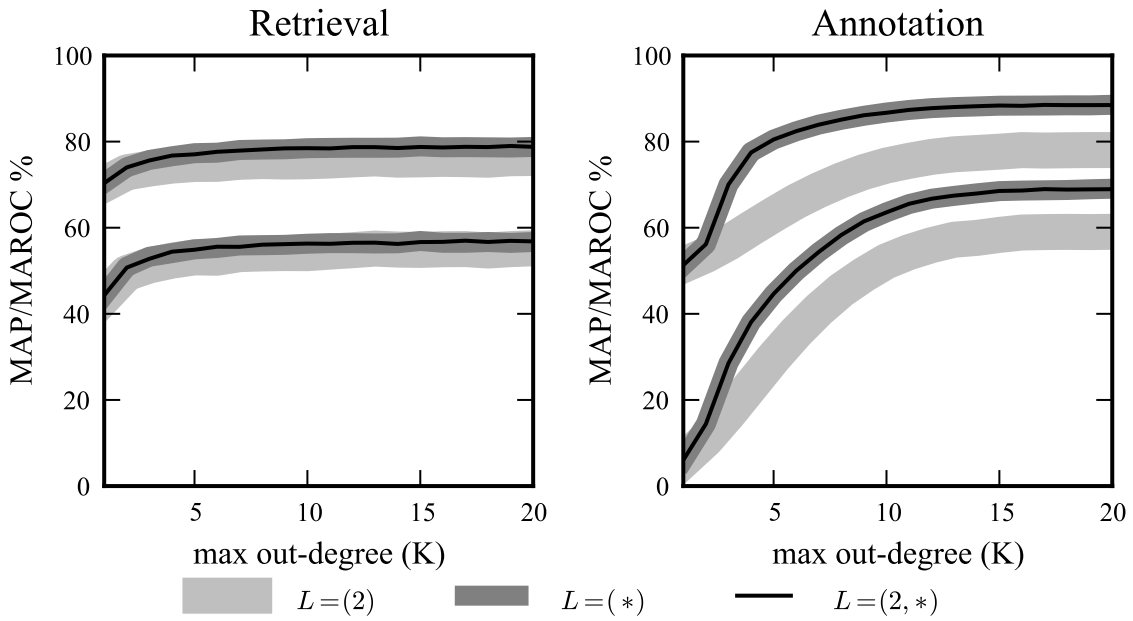


Figure 3.6: Effects of pruning outbound edges to nodes' K nearest neighbors using the *Soundwalks* dataset. Both MAP (lower) and MAROC (upper) values are plotted as a function of K and averaged across $n = 100$ trials with half the tagging data missing.

From these plots, we can see that, in some cases, as in annotation on the *Soundwalks* dataset, using shortest paths performs worse than the baseline case, likely because known sound-to-tag links are being circumvented in favor of paths that use acoustic similarity. However, $L = (*)$ seems to perform marginally better than $L = (2)$ for the case of retrieval. To account for this difference, we can see that prioritizing direct links, as in $L = (2,*)$, performs best. $L = (2,3,*)$ is a special case, as it produces higher MAP/-MAROC, corresponding to its better performance in the last 75% of results, but it initially performs more poorly at annotation, which may be undesirable (if, say, we were to annotate with the first N highest-scoring tags).

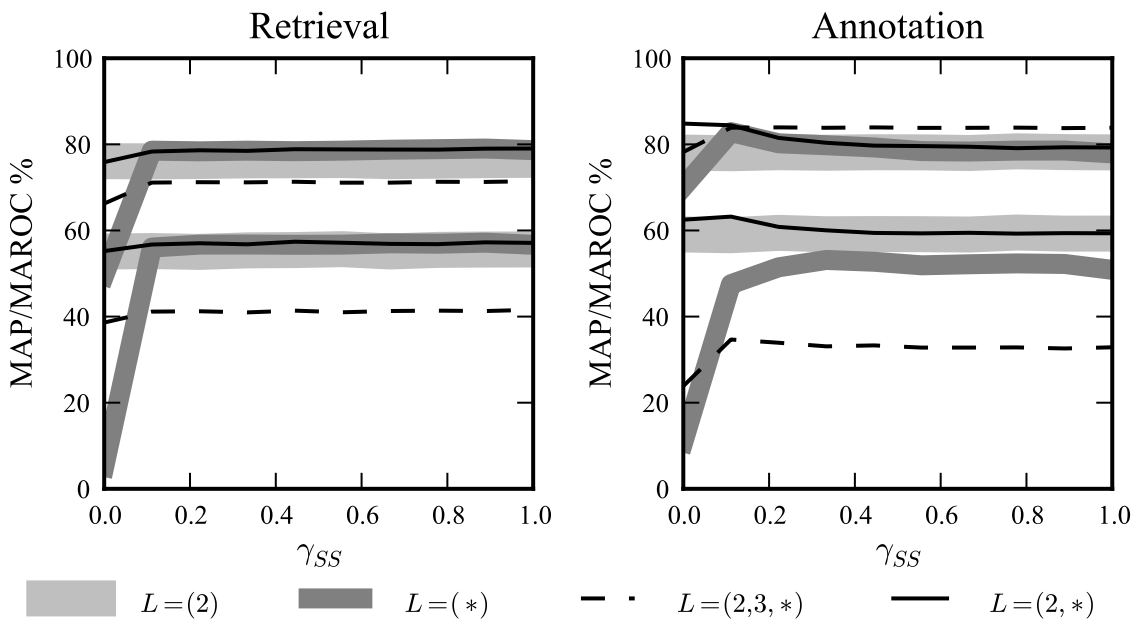


Figure 3.7: Effects of varying γ_{SS}, the global weight multiplier for sound-to-sound edges using the *Soundwalks* dataset. Both MAP (lower) and MAROC (upper) values are plotted as a function of γ_{SS} and averaged over $n = 50$ trials with half tagging data missing.

For the *Freesound* dataset, for which we provided no sound-to-sound links, we can see that the $L = (*)$, and optionally $L = (2,*)$, methods can assist in ordering the last half of results. This improvement is likely because, for annotation (and analogously for text-based retrieval), a sound can be annotated with additional tags from those sounds with which it shares a few tags. Of course, in some use cases, this increase in performance may not be worth the extra query time. Note that $L = (2,3,*)$ would behave the same as $L = (2,*)$ in this case, as no sound-to-tag paths with an odd number of nodes exist in a network containing no sound-to-sound edges.

3.4.4 Edge pruning

To test the effects of limiting search to nodes' K nearest neighbors, we first constructed a network as described in Section 3.1 using the *Soundwalks* dataset, with sound-to-sound and sound-to-tag links, but with half the tagging data missing at random. For $K \in \{1, 2, ..., 20\}$, we then annotated with each sound and retrieved with each tag, testing relevance against the original tagging data. For each value of K, we averaged performance metrics over 50 random trials for a total of 1000 trials per query type. As shown in Figure 3.6, it is only when $K < 10$ that significant losses in MAP/MAROC can be seen, suggesting that edge pruning can drastically improve query time without having significant effects on performance, as $10 \ll |E|$.

3.4.5 Weighting edge classes

Figure 3.7 demonstrates that, for the *Soundwalks* dataset, there is a clear shift in performance at $\gamma_{SS} \approx 0.2$. For $\gamma_{SS} < 0.2$, acoustic weights are used as the primary source of similarity information at the expense of known tagging data. For the case of annotation, there appears to be a slight increase in MAROC for $L = (*)$ at this point. For $L = (2, *)$, there is a slight increase in performance for $\gamma_{SS} < 0.2$, which suggests that the system performs slightly better when tagging data is used for direct links, but acoustic similarity, rather than co-occurrence of tags, is primarily relied on when no direct, 2-node links exist.

Chapter 4

SONIC CARTOGRAPHY

To address the challenges of organizing large soundscape archives mentioned in Chapter 2, a number of web-based sound maps have appeared within the last decade, perhaps beginning with Stanza's *Soundcities* [A.68] in 2000. Many of these tools are mash-ups with online mapping tools, having an interaction style that can be described as "point, click, and listen," where sounds are represented visually on a map as homogenous place marks.

While this type of representation allows one to explore a location's distribution of recordings and provides a convenient method of exploring large, geo-tagged audio archives, it perhaps lacks a fluidity that allows easy exploration, as the place marks rarely indicate any features of the sounds they represent other than location, and it may be necessary to listen to several long recordings before finding one that is relevant or interesting to a listener. To use a visual metaphor, it may be equivalent to clicking on individual map markers to view photographs of a city rather than the more continuous exploration allowed by satellite imagery or photo-stitching tools such as Bing Maps or Google Street View. Of course, photographic media (including satellite imagery) have the advantage of natural spatial properties, which are more difficult to represent with point-source recordings. Nevertheless, being provided with a continuous visual flow of multiple observations makes the process of viewing visual maps highly interactive without explicit manipulation, a strength that would be of great benefit to any soundscape mapping project.

The inclusion of acoustic information in cartography has many advantages, most importantly those involving the depiction of *activity information*: that which is highly time-dependent. If a viewer or listener is primarily interested in cultural information, for ex-

ample, rather than relatively stationary geological features, the sounds of human activity (anthrophony) can be very informative. Truax explains the topic as follows:

> Although the sound wave reflects every detail of motion of its source, its travel through an environment—reflecting from and being absorbed by all objects—is influenced by the general configuration of the environment. In a sense, the sound wave arriving at the ear is the analogue of the current state of the physical environment, because as the wave travels, it is changed by each interaction with the environment. Whereas vision allows us to scan an environment for specific detail, hearing gives us a less detailed, but more comprehensive, image of the entire environment in all directions at once. [2]

This suggests that while visual scanning allows a detailed view of specific features in an environment, it can gloss over activity that occurs over large scales. For a concrete example, we may not be able to see past the facade of a building, but we can often hear loud activity behind it. As another example, a low-flying airplane can often be heard quite loudly within miles of its flightpath. Several existing practices exist for plotting contour maps of complex loudness and annoyance models, akin to those proposed in [1], but this chapter will primarily focus on those maps that emphasize the recordings themselves, rather than acoustic measurements, on top of a geographic map for purposes of reflection.

Some sound archives, such as *Radio Aporee* [A.54] and *Freesound* [A.84], have introduced alternative ways of listening and searching, such as automatic playlist generation through random walks or folksonomy-based tag searching. Section 4.1, as reported in [86], will examine 95 different sound maps on the web, starting in 2000, categorizing them by their means of interaction, associated multimedia and metadata, and content curation, speculating on what future directions might exist in each of these areas. A complete list of these sound mapping projects is available in Appendix A.

It is also with these considerations that Section 4.2 will present the development of a new type of sound map, generically called *Soundwalks*, which attempts to extend the amount of geospatial information available to users by communicating acoustic information in a rich, interactive framework that will allow users to quickly gain insight into the contents of their soundscapes.

4.1 Interactive sound maps

4.1.1 Interactive features

1. Map embeds: Of the 95 geo-tagged sound archives listed, 80 were found to at least have some sort of point-click-and-listen interaction. Some of these are simply implemented as blog entries with single-marker embedded maps, presented more as a story of recording, focusing on the prose descriptions of the process as opposed to geographic relationships between sounds. However, most of these tools exist as full web sound maps, with 62 commercial mapping embeds and 5 using the Creative Commons-licensed, crowdsourced mapping service OpenStreetMap [87]. Although some of the other maps may use more sophisticated or well-designed custom visual layers, the prevalence of map/marker mash-ups is encouraging, as it demonstrates the relative ease of developing these maps, showing that they are an accessible form of archival and sharing.

2. Automatic Play and Advancement: One simple factor that can hinder interaction with a sound map is the number of clicks necessary to explore it. Only 23 maps marked with *autoplay* automatically begin playing sounds upon clicking their markers rather than having to initiate an embedded audio player, or worse, being redirected to a separate website where the sound can be played. Although fast-paced clicking is perhaps not the mindful or reflective style in which some authors wish

their listener to engage, being able to smoothly shift from one sound to another can make the resulting soundscape much more seamless and allow for quick comparison of acoustic properties between locations.

Another possibility for automatic play, referred to as *autoplay next*, is automatic advancement to other sounds on completion, featured in maps such as the *Montréal Sound Map* [A.45], *Favourite Sounds* [A.19], the *Brussels SoundMap* [A.10], and *Cartophonies* [A.12]. This feature is most useful for maps that include sound sequences and paths, but even queueing random walks between neighboring recordings, such as in *Radio Aporee* [A.54], allows a listener to sit back and actually listen, using the map as a visual reference rather than dedicating significant attention to mouse movements and interaction with audio widgets.

3. Sequences and paths: Six of the maps listed allow temporal sequencing of sounds. Sound sequences are especially interesting for the recording of the soundwalk. Some field recordings taken during soundwalks involve several segregated sound files. Being able to both initiate a sequence of sounds and have the player automatically advance to the next in the sequence allows the author to relate a sonic narrative. On some sound maps, such as *Listen to Africa* [A.32], knowledge of the path the recordists travelled and inspection of the geographic distribution of the place-marks is enough (they closely follow a coastline), but in dense archives such as *Freesound.org* and *Radio Aporee* (which provides this feature, along with six other maps), it can sometimes be necessary to organize a sequence explicitly.

   An obvious extension of the sequencing feature is to visually display connections between the sounds through GPS traces. In the simplest case, since most audio archives record the time and date of recording, this can be as simple as drawing line segments between sounds of the same recording session or soundwalk. In more de-

tailed settings, continuous GPS traces from external devices can be used, which can easily be obtained from most smartphones. In retelling the story of a soundwalk, in particular, these GPS traces convey important information about the walk, such as the moment-to-moment decisions made in improvisational exploration, paths left silent, or an overview of the spatial properties of a pre-composed soundwalk. These traces are especially important in the presence of lengthy (e.g. several hour) non-stationary recordings that are not segmented. As of yet, no web sound map seems to address the issue of conflating an entire, continuous mobile field recording onto a single point.

4. Mixing: Dynamic mixing of multiple field recordings is a particularly interesting pattern supported by only five of the maps, namely *Cinco Cidades Soundmap* [A.13], *Tactical Sound Garden* [A.80], *Favourite Sounds*, *soundingD SoundMap* [A.69], and *Radio Aporee*. Most of these tools allow for manual selection of several recordings to mix, which potentially allows the map itself to be a live performance or compositional tool, likely of interest to phonographers. One potential artifact (or affordance, depending on one's outlook) of these strategies is that directly mixing multiple soundscapes can result in a cacophony of overlapping acoustic streams that cannot be segregated psychoacoustically. Using Schafer's terminology, this can result in obfuscation of keynote sounds, not to mention interrupted signals and soundmarks.

5. Mobile applications and live streaming: Two additional patterns, supported by only a few platforms, include the presence of mobile applications and the mapping of live audio streams. *Audioboo* [A.5], a popular audio-blogging site, initially contained a sound map of recordings as its landing page, though they have since focused on a more social networking style, where the geographic metadata has un-

fortunately been relegated to the page view for individual sounds. The enterprise *SoundCloud* [88] can also provide GPS metadata when recordings are uploaded from its mobile application, but as of yet, besides custom pages that position them, it has not explored audio mapping as an official facet of its business. Two other non-commercial tools, however, do provide mobile applications, including the soundscape affect research project *Sound Around You* [A.64] and *Radio Aporee*'s recent mobile version.

Live streaming is an interesting modality that has existed for some time, and though only three of the listed platforms [A.33, A.34, A.50], aggregate multiple streams on a single map, a live stream does provide an excellent sense of a location's temporal activity patterns. *Listening to the Deep Ocean Environment (LIDO)* [A.33], in particular, provides real-time spectrogram of its live streams. Many other solitary live audio streams (mostly focused on bioacoustics applications) exist and could potentially be aggregated on a map as well.

4.1.2 Multimedia and metadata

1. Other forms of media: While reduced or musical listening approaches to the soundwalk may deemphasize the context of a sound stream, providing additional media related to a recording can be quite helpful in recording its experiential context. 16 of the maps provide one or more images alongside the recordings, and three of those also provide video content. These additions are especially common amongst field recording blogs. *Xeno-canto* [A.93], *LIDO*, and *Freesound.org* all provide spectrograms, which could potentially be interesting data to integrate more seamlessly into a map. The various sound maps of the *London Sound Survey* [A.35, A.71, A.83] also provide additional layers, such as historical maps.

2. Text description: 14 of the maps provide each sound with folksonomy-based single-

term tags, some of which allow browsing or searching by these tags. 28 of the maps provide segregated categories or taxonomies, some in particular being inspired by the taxonomy first proposed by Schafer. More generally, 70 maps provide some description of the sounds being played, though often these can be limited to descriptive filenames. 12 maps provide all three descriptors. Social media repositories tend to implement folksonomy-based tagging, but explicit categorization can be quite useful, especially when a comparative study is desired. While not present on all maps, this type of metadata is one step away from providing a more investigative experience for listeners. Content-based search, such as techniques used in music information retrieval or speech analysis are also obvious candidates, though as of yet unexplored in these maps.

4.1.3 Content curation

1. Community involvement: The 95 sound maps also vary drastically in how they curate content. 69 of the maps are either now closed or limited to the contributions of the authors. The remaining 26, however, allow some sort of community contribution, varying from automatic submission interfaces to explicit request via email. *Sound Around You* has an interesting approach from a research standpoint, where users are not only able to upload their sounds from mobile devices, but are also able to comment on the soundscapes' affective properties. Crowdsourced data, of course, brings up obvious issues of licensing restrictions. It is fortunate, then, that the larger projects, *Radio Aporee*, *Freesound.org*, and *SoundCloud* allow users to choose between various licensing schemes, including Creative Commons options.

2. Other sources of audio: While field recordings largely (or solely) make up 76 of the listed projects, there are some sound maps that focus explicitly on music or speech, such as some of the mapping projects by The British Library. *Audioboo* likely has

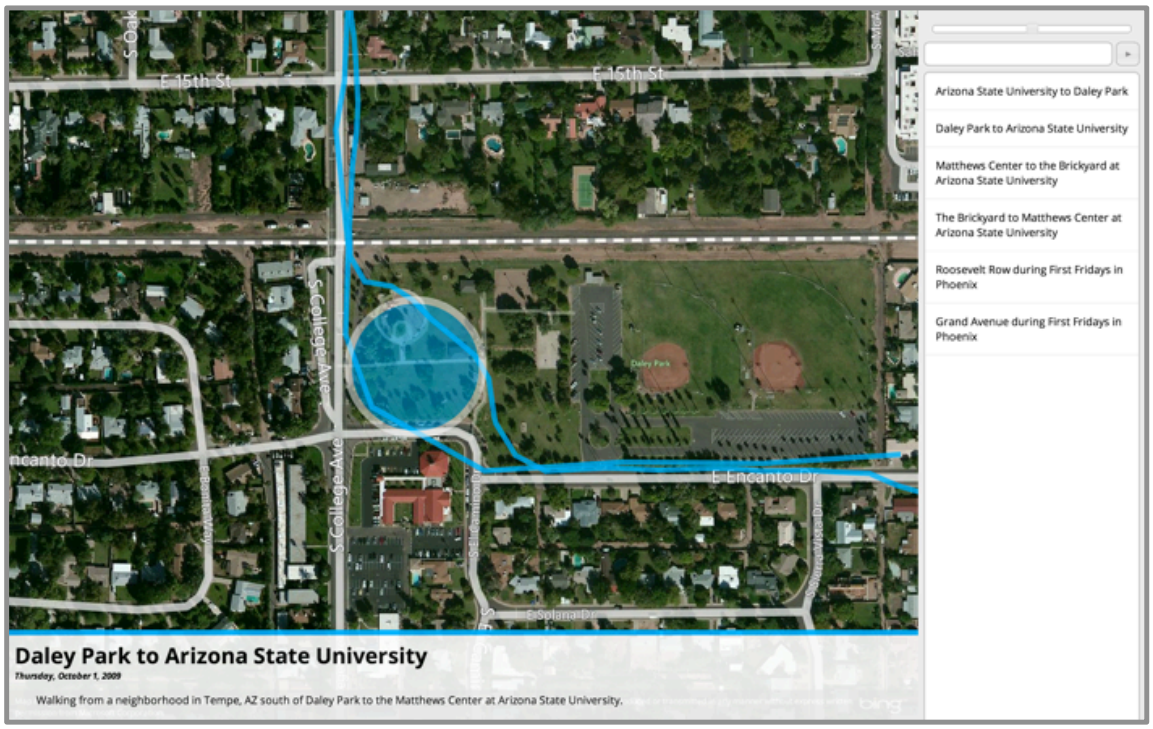


Figure 4.1: An example of the *Soundwalks* interactive synthesis interface. The cursor is situated between two paths, where a mixture of the two field recordings will be synthesized.

a similar focus on speech, with its branding as an audio blogging tool. Additionally, a few of the soundmaps focus at least in part on historical data, including the soundmaps of the *London Sound Survey* and efforts by The British Library, the latter of which include recordings from historical archives.

4.2 Continuous playback with *Soundwalks*

Soundwalks, a new soundscape mapping tool, attempts to elaborate on some of the features mentioned in the previous survey of web sound maps. Most importantly, it addresses issues related to uninterrupted exploration of the soundscape and visualization of the soundwalk path. As will be discussed in Chapter 5, uninterrupted exploration is

addressed by parametrically mixing between multiple continuous field recordings.

4.2.1 Application architecture

In its current form, Soundwalks is a web application intended for personal archival and reflection upon acoustic journeys, rather than a large, crowdsourced application. Soundwalks works as an HTML5 application that uses the WebSocket API to communicate to a local server running "pyo," [89] a Python-based modular synthesis library. Optionally, the server can also route messages from the browser as OSC [90] messages to be used by other synthesis applications or interfaces such as PureData [91], Max/MSP [92], and ChucK [93].

4.2.2 Interaction through scrubbing

The user interface shown in Figure 4.1 is sparse, showing a list of soundwalks with associated titles and descriptions that move the map to their coordinates when clicked. Every full GPS path, imported via GPX from a mobile application or dedicated GPS receiver, is shown. As the user drags a cursor across the map, regardless of whether or not a path is underlying, a continuous soundscape is played. This soundscape is a re-synthesized audio texture resulting from a parametric blend of clips from each soundwalk within a defined radius of the user's cursor. This mix is constructed through a variation of the wavelet tree learning technique in [94], which attempts to create variations on audio textures while retaining their structural properties as opposed to more common granular synthesis clouds or concatenative synthesis, as will be discussed in Chapter 5. The soundwalks closest to the cursor are favored in this re-synthesis process.

4.2.3 Implications of the synthesis model

Some benefits of a real-time mixing and scrubbing interface include the explorative navigation more characteristic of visual maps, where the same type of instant navigational decisions can be made, as listeners have constant, immediate feedback of their actions. Additionally, explicit inclusion of continuous GPS paths for each mobile recording allows viewers to better understand the geographic context and sequence of navigational decisions made by the recordist in the moment. However, it should be noted that this method is by no means meant to be an end-all-be-all solution to acoustic mapping. As has been detailed in the survey of existing sound maps, there are a wide variety of sonic cartography projects, each with their own theoretical backings, artistic statements or intent, and preferred listening style.

Chapter 5

MULTISOURCE SOUNDSCAPE SYNTHESIS

To facilitate a more responsive, interactive experience in exploring soundscapes, *Soundwalks* uses a method that allows users to continuously morph between sound sources. While a system could repetitively loop sounds nearest a user's cursor, it may take some time to listen to each sound in its entirety, and the experience may be aesthetically displeasing. To this end, we wished to satisfy the following requirements:

1. Responsiveness: To facilitate realtime exploration, users should be able to hear a change in the soundscape quickly, if not immediately, after moving the cursor.

2. Realism: The soundscape should be realistic, with few artifacts such as clipping, interrupted sound events, etc.

3. Source relevance: The soundscape should most closely resemble the sounds recorded nearest the cursor.

To support these requirements, we have developed a technique of synthesizing continuous, non-repetitive sound textures from several example recordings. Several methods have been devised for the purpose of synthesizing realistic sonic textures (see [95] for a recent survey.) In many cases, these methods are inspired by techniques designed for the analogous problem of visual texture synthesis, in which some important goals include reducing the tiling effect, making any tiling boundaries between texture components invisible, and preserving the overall structure of the original image [96]. For sound textures, these goals can be thought of as minimizing the number of transients introduced by the synthesis and preserving the audible, qualitative characteristics of the

original sound sources. In fact, it is perhaps most important to retain those psychoacoustic features of the original sound that characterize it as a separate auditory stream, as modifying these can either break the sound into two or more streams or merge it with other streams, which can be distracting and reduce ecological validity. [97] discusses some of these issues, classifying undesirable properties as *echo*, *cutoff*, and *repetition*.

5.1 Concatenative synthesis

While several methods for sound texture synthesis by example have been developed, few have been designed for the purpose of parametric mixing between *multiple* sources. [98] and [97] both describe concatenative synthesis methods where sound grains [99; 100] (or short clips) are randomly selected from a single source sound and used to construct a new random texture using overlap-add. In [98], clips are overlapped using Gaussian envelopes with 15% overlap and grains are sequenced in a stochastic process that attempts to match their beginning and ending 15% of samples according to similarity. No such ordering criteria are presented in [97], which uses cosine windows with half overlap and instead focuses on stochastic variations in sub-clip features, such as amplitude variations, and finding the optimal sub-clip duration, which, through listening tests, they find to be approximately two seconds in length.

One application of concatenative synthesis discussed in [101] is that of cross-corpus synthesis, where sound grains from one corpus are used to re-synthesize an input sound with different grains. For example, Schwarz uses a violin to re-synthesize human speech or an electronic music piece.

5.2 Wavelet tree learning

[102] demonstrates a technique for visual texture synthesis that is often used as a benchmark for subsequent methods. This algorithm uses a multi-resolution, pyramid-

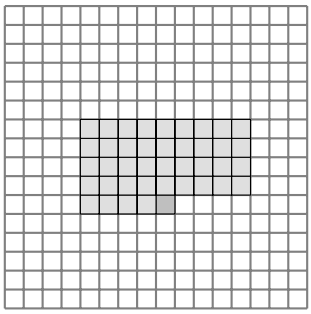


Figure 5.1: An example 9×9 neighborhood in an image texture. Note that pixels that have yet to be determined (i.e. are later in the scanning order) are not part of the selected pixel's neighborhood.

based approach for texture re-synthesis. First, a texture is modeled as a Markov random field, where each pixel is paired with an associated L-shaped neighborhood of the surrounding $K \times K$ pixels, excluding those pixels which have a scan order greater than itself (see Figure 5.1 for an example). One can then construct a new texture by randomly selecting a pixel from the source image and then choosing each new pixel in a row-major scanning order according to that which has the most similar neighborhood to those pixels already selected, possibly with a randomization factor for the sake of diversity. In the case of [102], this is done by selecting from a set of pixels whose neighborhoods have a similarity to the previously-selected pixels above some threshold.

While a single-resolution scheme captures certain small features quite well, many perceivable features in an image texture exist at a scale larger than the neighborhood size. To account for this, Wei and Levoy use a pyramidal structure where, for a source texture of size $N \times N$ where $N = 2^L$, the source texture is down-sampled between 1 and $L-1$ times. From here, the same selection process occurs consecutively for each level, beginning with the level corresponding to the lowest resolution.

A technique similar to this was devised by Bar-Joseph, et al. [103], where instead of downsampling using a Gaussian pyramid, a multi-resolution approach was achieved

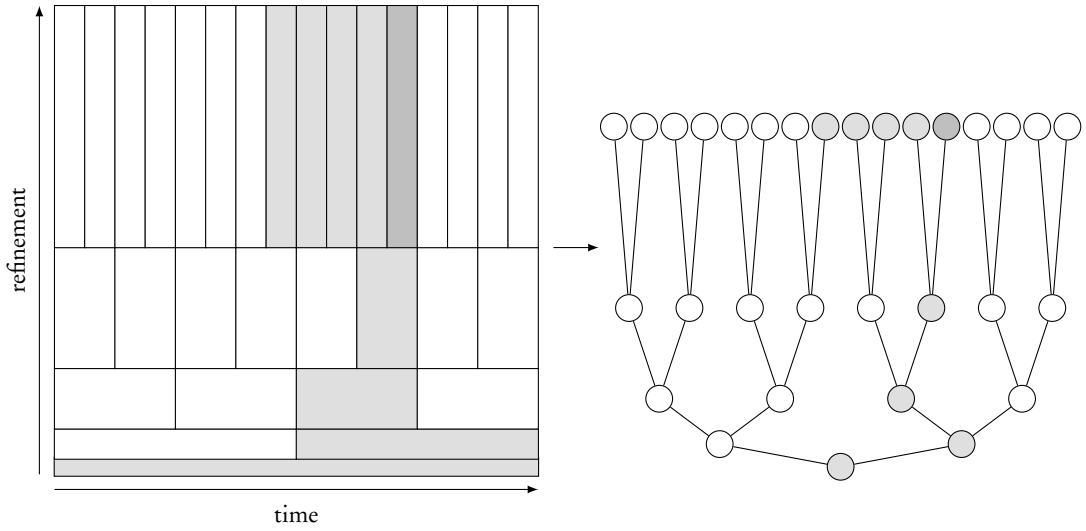


Figure 5.2: An example neighborhood for comparison with candidates nodes, where $K = 5$ predecessors are considered. The left graphic displays the scalogram that results from a multi-resolution wavelet decomposition of a one-dimensional time-varying signal. The right graphic shows the same coefficients represented as nodes in a binary tree on which the wavelet tree learning algorithm operates. In both images, a selected node and its neighborhood are shaded.

through a multi-resolution discrete wavelet transform (DWT). In this algorithm, an image is convolved with a particular wavelet (in this case, the Daubechies wavelet with 5 vanishing moments, D10), down-sampled by 2, and convolved again. This process occurs several times to construct a multi-resolution analysis (MRA) tree, a tree where each node represents a DWT coefficient. From there, a similar candidate selection takes place, but neighborhood similarity is calculated over the domain of DWT coefficients rather than down-sampled pixels. The process of creating an MRA tree is demonstrated in Figure 5.2.

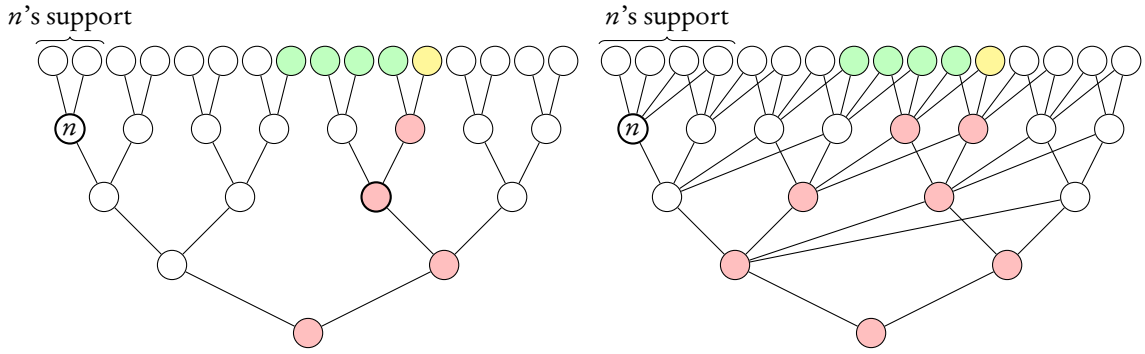
This technique inspired a new method of sound texture synthesis, described in [94], where the authors apply the same multi-resolution analysis for a one-dimensional signal in a technique they refer to as *wavelet tree learning*. The process of wavelet tree learning

involves creating a new MRA tree from one devised from a source signal (in this case a monophonic audio recording) that shares certain local properties with the original sound texture. Starting with the root node, a candidate set of nodes is constructed from which a coefficient will be selected at random. The candidate set is initialized with the set of all coefficients at the same level in the tree and is then culled according to those which best preserve temporal ordering constraints (a neighborhood of the previous K coefficients on the same level) and hierarchical structure constraints (a neighborhood of L ancestors of the current node). After the tree has been constructed, the original signal can be reconstructed through successive applications of the inverse DWT.

5.2.1 Applicabality to texture morphing

Wavelet tree learning has shown to be quite effective for certain acoustic and visual textures. However, the algorithm tends to produce poor results when transient or deterministic events, such as bells or horns, are present in the original recording. Wavelet tree learning makes the assumption that the source signal has some homogeneity within each level of decomposition, so performing the algorithm on an audio clip that includes deterministic events or objects will frequently interrupt them, causing strange splits and scatterings in visual textures and transients and cutoff in acoustic textures. When effort has been taken to remove these events, such as in the case of *TAPESTREA* [104], the texture can be successfully synthesized, as the source material is likely to resemble a single texture without sound events.

Additionally, wavelet tree learning can be quite computationally expensive. Performing the multi-resolution wavelet tree decomposition can be done in $O(n \log(n))$ time, but performing the learning can take many cycles, having a complexity of at least $O(n^2)$ time, depending on the number of K predecessors present in a coefficient's neighborhood. Even in the case of an efficient low-level implementation, this can take longer than



(a) D2/Haar (b) D4

Figure 5.3: An example MRA tree of a one-dimensional signal demonstrating how a network would be structured with full support. For an example node, predecessor nodes are colored green and potential ancestor nodes are colored red. (a) displays the support of a D2 or Haar wavelet, which WTL implicitly assumes, while (b) shows the support of a longer wavelet, D4. For a neighborhood with $K = 5$ predecessors, one can see how taking into consideration a wavelet's true support can greatly increase the neighborhood size.

realtime. Perhaps by splitting a recording into sufficiently small windows, each of which are re-synthesized independently, this process can operate in realtime. This technique will be discussed in Section 5.4.2.

Wavelet tree learning also makes a certain assumption about the support of wavelet coefficients that can cause the introduction of transients, even in relatively stationary textures. Specifically, wavelet tree learning constructs a binary tree from the coefficients of the multi-resolution decomposition. While each stage of the decomposition does downsample the signal by a factor of two, this does not mean that a coefficient has only been calculated from the two coefficients that become its children in the MRA tree. Rather, most wavelets, other than the D2 / Haar wavelet with filter length of two, are calculated from more samples. [104] and [94] use the D10 wavelet, which has a filter length

of ten coefficients (five vanishing moments). In this case, rather than a binary tree, to truly model the support of each node, it would have five immediate ancestors and ten children. Figure 5.3 demonstrates this difference. In forming a new MRA tree, nodes are selected based upon matching neighborhoods which include only single ancestors per node, which can cause certain spectral and temporal relationships in the original sound to be broken. One implementation of WTL, however, has suggested that considering ancestor relationships does not particularly affect the overall quality of the synthesized results anyway [105].

While Dubnov et al. do not specifically mention the possibility of mixing two or more sounds, the method is explored for the purpose of video texture mixing in [96], where candidate nodes are selected from more than one MRA tree. For a parametric approach, one can prefer nodes of the respective tree in the stochastic candidate selection, but the paper mentions that this can lead to a certain "lock-on" effect, where early selection of a node from a particular sound can lead to large chunks sourced only from that sound.

5.3 Comparisons

Subjective evaluations of several texture re-synthesis techniques were compared in [104], including a random overlap-add approach comparable to that in [97], three methods that sequence the sub-clips by minimizing differences between MRA trees of sequential clips, and two methods that sequence clips according to similar RMS levels. After finding optimal tunings for each algorithm through separate sets of subjective listening tests, the author reports that subjects found the random overlap-add method to be most perceptually convincing, though the source sounds tested were rather stationary textures, which may have made structural properties, which WTL is designed to retain, less important to listeners.

5.4 New morphing techniques

We have focused on comparing four methods for parametric sound texture morphing, inspired by existing work in concatenative synthesis and wavelet tree learning. Two methods use concatenated grains, and two methods use a version of WTL where the grains in a concatenative sequence are first re-synthesized. For each type of re-synthesis, one version creates a single stream of grains of heterogenous origin—stochastically choosing grains' sources, weighted according to the desired mix—and another version produces streams of grains of homogenous origin, one per source, and mixes the sequences accordingly.

5.4.1 Concatenative synthesis

In all cases, streams are composed of sequences of grains, $s = (g_1, g_2, ...g_n)$, each of which has a duration, $Dur(g_i)$, source recording $Src(g_i)$, beginning time offset within the sequence, $Pos(g_i)$, and beginning time offset within the source, $SrcPos(g_i)$. Grains are always concatenated such that they overlap by half the duration of whichever grain is shorter, i.e.

$$Pos(g_i) := Pos(g_{i-1}) + Dur(g_{i-1}) - Fade(g_i, g_{i-1}), \tag{5.1}$$

where *Fade* is the amount of overlap (the length of the crossfade) between grains, defined as

$$Fade(g_i, g_j) := \frac{1}{2}\Big(\min\big(Dur(g_i), Dur(g_j)\big)\Big). \tag{5.2}$$

Overlapping grains are then finally mixed with a an equal-power crossfade. Duration and source position are defined stochastically, with

$$Dur(g_i) \sim \mathcal{U}(mindur, maxdur), \tag{5.3}$$

$$SrcPos(s_1) := \frac{1}{2}\Big(Dur\big(Src(g_1)\big) - Dur\big(g_1\big)\Big), \tag{5.4}$$

and

$$SrcPos(g_i) := SrcPos(g_{i-1}) + J \tag{5.5}$$

where J, the duration to "jump," forward or back is defined as

$$J := \alpha J_b + (1-\alpha) J_f, \tag{5.6}$$

$$J_b \sim \mathcal{N}\left(-Dur(g_i), maxdist\right) \qquad J_f \sim \mathcal{N}\left(Dur(g_{i-1}), maxdist\right), \tag{5.7}$$

$$\alpha := \begin{cases} \max\left(1, \dfrac{\left|SrcPos\left(g_{i-1}\right) + Dur\left(g_{i-1}\right) - C\right|}{maxdist}\right) & \text{if } SrcPos\left(g_{i-1}\right) \geq C \\ 1 - \max\left(1, \dfrac{\left|SrcPos\left(g_{i-1}\right) - Dur\left(g_i\right) - C\right|}{maxdist}\right) & \text{if } SrcPos\left(g_{i-1}\right) < C, \end{cases} \tag{5.8}$$

where C is the time offset in the recording corresponding to the point in its GPS path closest to the user's cursor. While $SrcPos(g_i)$ could easily be sampled from a gaussian surrounding C, this method sets up a mixture of gaussians that will minimize the amount of overlap between successive grains from the same source, thus avoiding repetition. α is selected such that the direction (forward or back in the source recording) will tend toward the point closest to the cursor if it meanders too far away from *maxdist*. This synthesis technique then has three free parameters: *mindur*, *maxdur*, and *maxdist*.

5.4.2 Windowed wavelet tree learning

In windowed WTL, the same sequence model described above is used, but each grain undergoes an additional step of re-synthesis using WTL. WTL has three additional free parameters, k, p, and *maxlevel*, which define the number of predecessor MRA nodes to use for comparison ($\lfloor k * 2^l \rfloor$ predecessors, where l is the level of the node in the tree), at what percentile of ancestor and predecessor similarity to threshold candidate nodes, and at what level of the MRA tree to stop the wavelet tree learning process [104].

5.4.3 Single- or multi-stream synthesis

The sequencing model described above assumes that there is a single possible source per stream. In the multi-stream models, these sequences are then mixed according to the distances from the cursor to the points closest to it on the recordings' GPS paths. In the single-stream models, each grain's source is randomly selected from this distribution, so it is necessary to adjust the logic that describes the jump distance, J:

$$J := \alpha J_b + (1-\alpha) J_f, \tag{5.9}$$

$$J_b \sim \mathcal{N}\left(-Dur(g_i), maxdist\right) \qquad J_f \sim \mathcal{N}\left(Dur(g_{i-p}), maxdist\right), \tag{5.10}$$

$$\alpha := \begin{cases} \max\left(1, \dfrac{\left|SrcPos\left(g_{i-p}\right) + Dur\left(g_{i-p}\right) - C\right|}{maxdist}\right) & \text{if } SrcPos\left(g_{i-p}\right) \geq C \\ 1 - \max\left(1, \dfrac{\left|SrcPos\left(g_{i-p}\right) - Dur\left(g_i\right) - C\right|}{maxdist}\right) & \text{if } SrcPos\left(g_{i-p}\right) < C, \end{cases} \tag{5.11}$$

where g_{i-p} is the most recent grain from the same source, i.e.

$$p = \min\left\{j : Src(g_{i-j}) = Src(g_i), j \in \mathbb{N}\right\}. \tag{5.12}$$

5.5 Evaluation

5.5.1 Testing set

The morphing techniques discussed allow parametric mixing between multiple sound sources associated with various geographic paths. For the purpose of evaluation, we have focused on a specific case where three recordings of equal duration are associated with paths that make up the edges of an equilateral triangle. Each vertex of the triangle represents the end of one soundwalk and the beginning of another (see Figure 5.4). We then choose points within this triangle to affect the mixing distribution for each algorithm.

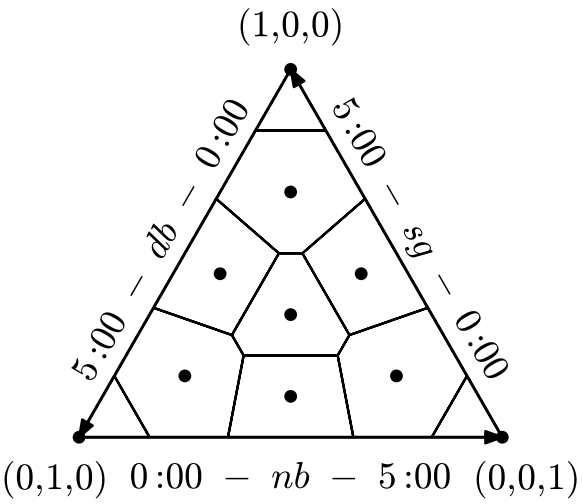


Figure 5.4: Synthesis coordinate space. *db*, *nb*, and *sg* represent different five-minute field recordings that meet along the triangle's vertices. The vertices (labeled with their barycentric coordinates) and equally-spaced points within are used as cursor positions at which to evaluate the texture morphing algorithms.

The three vertices and seven evenly-spaced points within the triangle have been selected for synthesis coordinates. This triangular parameterization was chosen over a simpler two-sound study to show the ability of the algorithms to handle more complex mixes.

For the source sounds, we have used three five-minute recordings. The recordings were down-mixed to monophonic 16-bit 44.1KHz versions and include

1. *db*, or "dog beach," a daytime recording at a dog beach, characterized by dog barks and yips, shouting owners, and surf;

2. *nb*, or "neighborhood birds," a dawn stroll in a quiet suburban neighborhood including sounds of footsteps, several birds, and a single passing car; and

3. *sg*, or "secret garden," a nighttime recording in a small outdoor courtyard on a university campus, isolated by surrounding buildings, including a cricket chorus, distant traffic, and an air conditioning unit.

5.5.2 Metrics

5.5.2.1 Perceptual convincingness

To evaluate the performance of the various algorithms, we have focused on two specific metrics that match our aforementioned requirements of realism and source relevance. First, we focus on the *perceptual convincingness* of a sound. For each synthesized version, we wish for it to sound realistic. Similar to studies performed in [97] and [104], we have asked human subjects to directly compare pairs of synthesized sound textures in a series of A/B tests with the simple prompt, "which sound is more realistic?" In addition to "Sound 1" and "Sound 2," we provide subjects with two more possible responses, including "Don't know" and "No difference." A pairwise comparison approach was chosen over an interval scale due to the lack of a well-defined minimum and maximum ground truth for realism. While a subjective measure of realism may be difficult to define, let alone quantify, these comparisons can give an approximate ranking of which synthesis techniques might be best to use in a production setting. One can imagine, for example, that a simple superposition of the three source sounds used for the mix might sound rather unrealistic, as several competing backgrounds are placed on top of each other. However, more complex mixes high in synthesis artifacts may also sound quite unrealistic.

5.5.2.2 Source relevance

Perceptual convincingness alone, however, would be an insufficient evaluation of a resynthesis algorithm, as it does not reflect how accurately the texture mimics properties of its sources. To account for this, we have also focused on a metric of *source relevance*. In this test, 10-second clips are extracted from the original sources from positions nearest to the coordinates used for synthesis. From two possible source recordings, the subject is

asked to evaluate which recording is more similar to the synthesized example provided. Each possible source recording is compared in a pairwise manner, making for three comparisons per synthesized version. Subjects are also provided options of "Equally similar" and "Don't know." This metric is used to evaluate which technique best retains the perceptual features it is trying to mix. Similarly, we can see how smooth each mixing process is by evaluating similarity to each of the source sounds as they are used more or less in the mix.

5.5.3 Crowdsourced survey

To obtain judgements, we used the crowdsourcing platform CrowdFlower.com. Subjects were limited to native English speakers to simplify instruction. Each task consisted of five random pairwise comparisons. Each comparison was scored by five unique workers.

Tests were performed in a three-stage process in order to reduce the number of responses necessary and allow each algorithm to be evaluated in terms of its most optimal tuning:

1. First, for each possible tuning, coordinate, and algorithm, five random versions were compared, making for a total of ten comparisons each (in total, 36,000 judgments). The most perceptually convincing sound of these five was promoted to the second stage.

2. Second, for each possible coordinate and algorithm, the remaining sounds were compared (1,800 judgements per algorithm). The most perceptually convincing sounds were then promoted to the third, final stage.

3. Lastly, All remaining sounds were compared for perceptual convincingness (3,900 judgments) and evaluated for similarity to their sources (600 judgements).

By comparing all possible sound pairs in the third stage, we were able to evaluate not only which algorithm scored highest for each coordinate, but we were also able to visualize how perceptual convincingness is distributed throughout the triangular coordinate space to see if it was considered more or less convincing when the mix becomes more complex.

Paid crowdsourcing platforms are common targets for automated bots and cheating among users for quick monetary reward, so in addition to the CAPTCHAs provided by CrowdFlower, several checks were put into place to ensure the integrity of the collected data. CrowdFlower uses the term "gold data" to refer to tasks with known outcomes used for validation. In every page of tasks presented to users, at least one task is golden, and users are required to first complete a "training page" of 5 golden tasks. Two types of gold were constructed: *identical sounds* and *synthetic tones*. For *identical sounds*, in the realism tasks, both sounds may be the exact same, requiring a response of "Don't know" or "No difference." In the source relevance tasks, the test sound may be identical to one of the possible sources provided. For *synthetic tones*, one of the sounds in each task may be a series of sinusoidal tones, which is unlikely to be considered more real than another sound, requiring that the other sound be selected as more realistic (or more similar). In this way, subjects cannot complete tasks by always selecting a null answer or by continually selecting sounds at random. Additionally, the sounds are presented using an audio player that has only a play/pause button and no option for scrubbing.

5.6 Results

5.6.1 Parameter tuning

The four mixing techniques, single-stream concatenative synthesis (*cs*), multi-stream concatenative synthesis (*csmix*), single-stream windowed wavelet tree learning (*wtl*), and multi-stream windowed wavelet tree learning (*wtlmix*) each require a number of param-

		(*mindur*, *maxdur*)					
		cs			*csmix*		
		(.5*s*, 1*s*)	(1*s*, 2*s*)	(2*s*, 4*s*)	(.5*s*, 1*s*)	(1*s*, 2*s*)	(2*s*, 4*s*)
maxdist	5*s*	2.68	4.27	4.83	3.81	4.16	4.29
	15*s*	2.47	4.60	5.16	3.63	3.84	4.27
	30*s*	2.85	4.15	4.99	3.80	3.91	4.29

		k					
		wtl			*wtlmix*		
		0.025	0.050	0.100	0.025	0.050	0.100
p	0.50	4.31	3.79	3.74	4.07	3.88	3.93
	0.75	4.01	4.18	4.18	3.90	4.03	3.99
	1.00	3.87	4.07	3.85	3.93	3.99	4.14

Table 5.1: Tuning results. Values indicate average preference for each combination of tuning parameters in terms of perceptual convincingness.

eters to be tuned in Stage 2. All methods require three parameters, *mindur*, *maxdur*, and *maxdist* to be tuned. To reduce the number of parameters required to tune *wtl* and *wtlmix*, which additionally require two more parameters, k and p, we have decided to use the resulting tunings that have highest average preference from *cs* and *csmix*, respectively. The *maxlevel* parameter for *wtl* and *wtlmix* was fixed at 10, corresponding to MRA nodes responsible for approximately twenty milliseconds.

Wherever multiple sounds are compared, the total preference of any sound, $pref(s_i)$, is

$$pref(s_i) := \sum_{s_j \in S, s_j \neq s_i} \frac{n_+(s_i,s_j) + \frac{1}{2}\left(n_=(s_i,s_j) + n_?(s_i,s_j)\right)}{n_+(s_i,s_j) + n_-(s_i,s_j) + n_=(s_i,s_j) + n_?(s_i,s_j)}, \tag{5.13}$$

where, over a set of sounds, S, against which s_i is being compared, $n_+(s_i, s_j)$ is the number of users who preferred sound s_i to sound s_j, $n_-(s_i, s_j)$ is the number of users who preferred s_j to s_i, $n_=(s_i, s_j)$ is the number of users who preferred s_i and s_j equally, and $n_?(s_i, s_j)$ is the number of users who could not provide a preference for either sound. In the case of tuning, S is the set of nine possible tunings for each coordinate and synthesis method. The resulting average preference values for each parameter tuning are listed in Table 5.1. Across all perceptual convincingness comparisons, an average of 3.9 of the five subjects agreed with the most popular answer, showing reasonable inter-subject agreement.

On average, it seems that longer grain lengths were preferred, which can be expected as there will be fewer transitions between time offsets and sources. Any longer maximum grain length, however, would reduce the responsiveness of the synthesis to user input. A moderate *maxdist* of 15 seconds was preferred, perhaps because the smallest *maxdist* incurred too much repetition, whereas a larger distance resulted in larger leaps in time within each recording, resulting in larger textural differences between grains.

In *wtl*, it seems that selecting from more predecessors ($p = 0.50$) and only requiring short-term similarity ($k = 0.025$) is preferred, whereas the opposite is true for *wtlmix*. The next highest score, however, is again at the least restrictive settings for candidate selection. As will be shown in Section 5.6.2, *wtlmix* is preferred over *wtl*, so it may be that *wtl* is so unconvincing that users express preference based on other criteria, such as preferring less discernible structural properties, which would result from more lax candidate selection.

5.6.2 Perceptual convincingness

Figure 5.5 shows the perceptual convincingness preference between methods, with preference for a specific algorithm defined as in Equation 5.13, where the algorithms

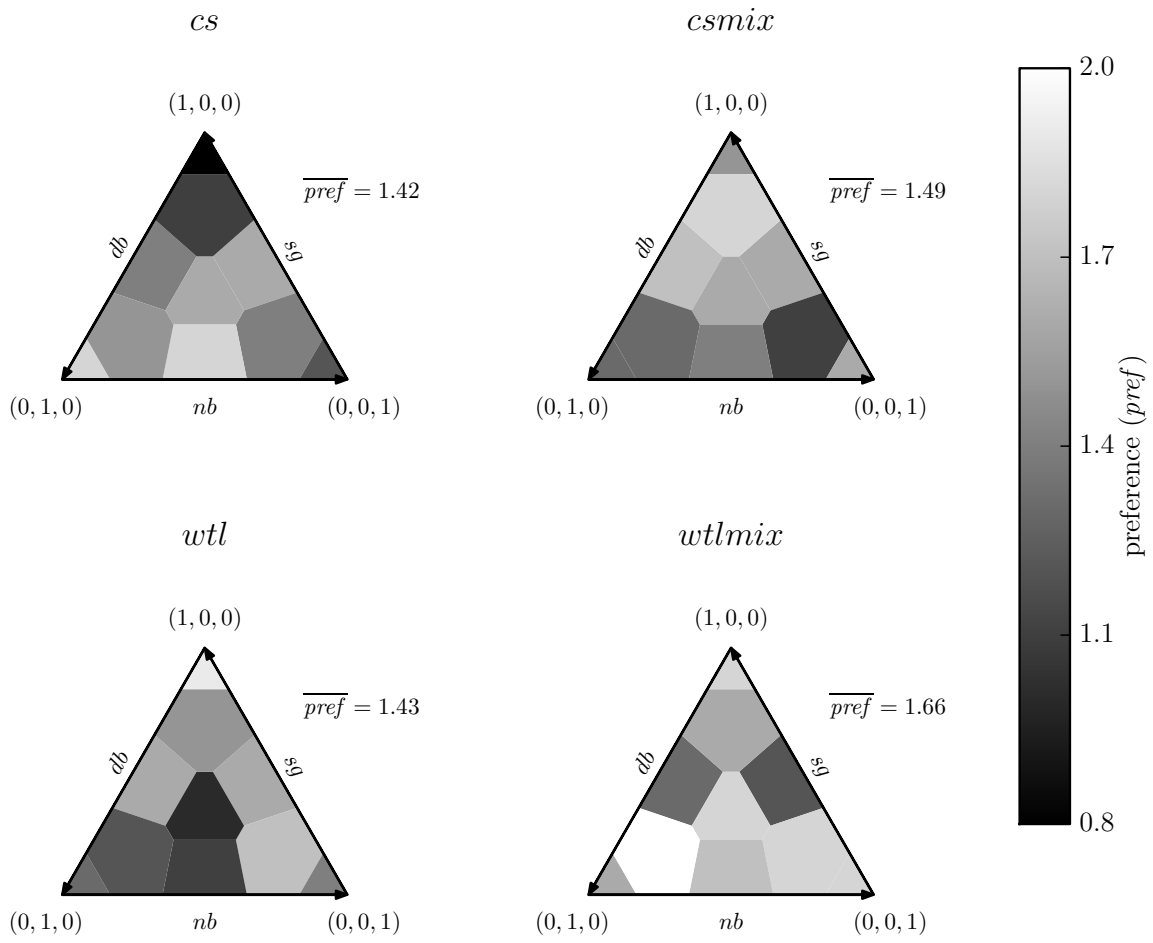


Figure 5.5: Perceptual convincingness rankings for each method. The shading of each voronoi cell corresponds to the average preference of the mixing technique given its respective synthesis coordinate.

themselves are compared rather than specific tunings. Each synthesis coordinate is represented by a shaded voronoi cell (the area of points closest to it). Figure 5.6 shows the preference between *coordinates*, holding the synthesis method constant. This plot shows how realism varies within the coordinate space for each method. We can see that on average, *wtlmix* was most preferred. Additionally, it seems that the multi-stream methods were preferred over the single-stream methods, perhaps due to fewer large textural

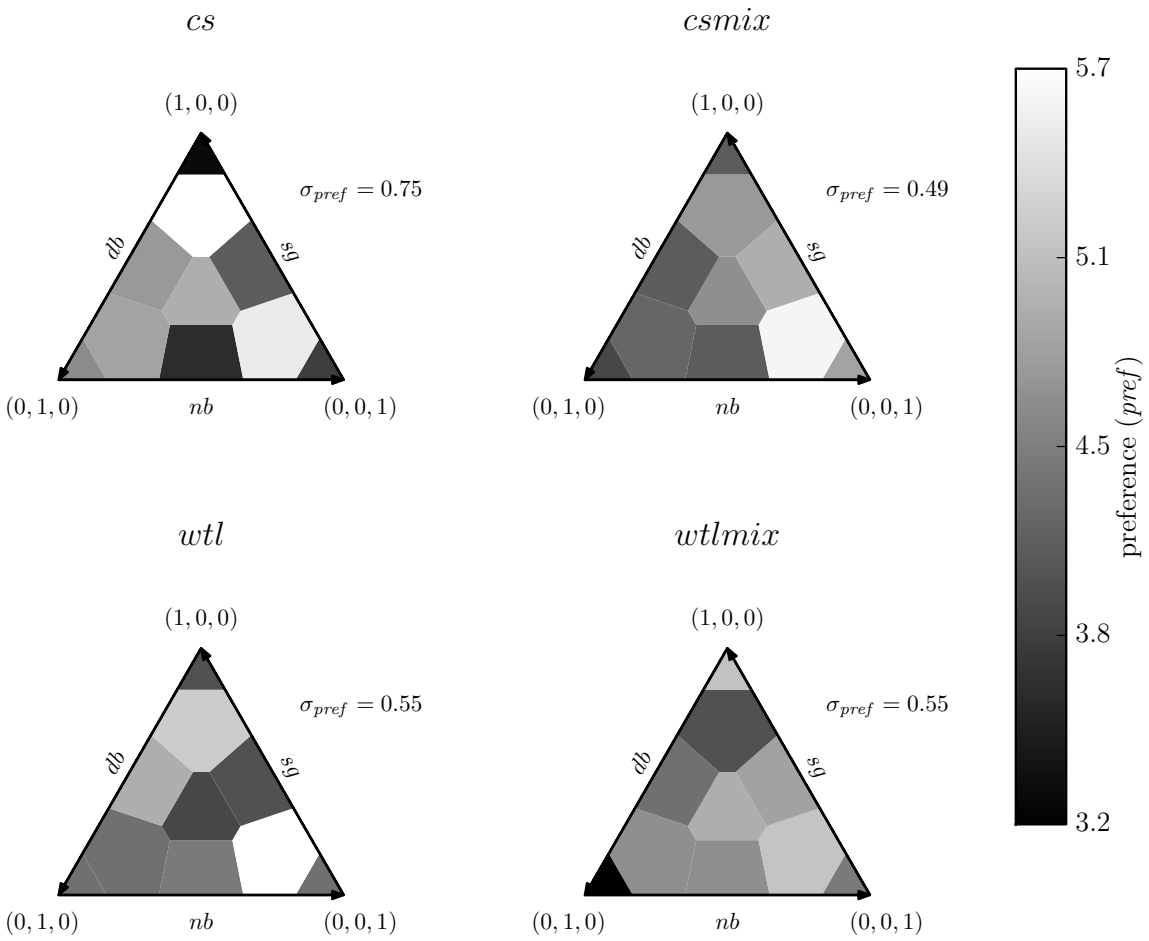
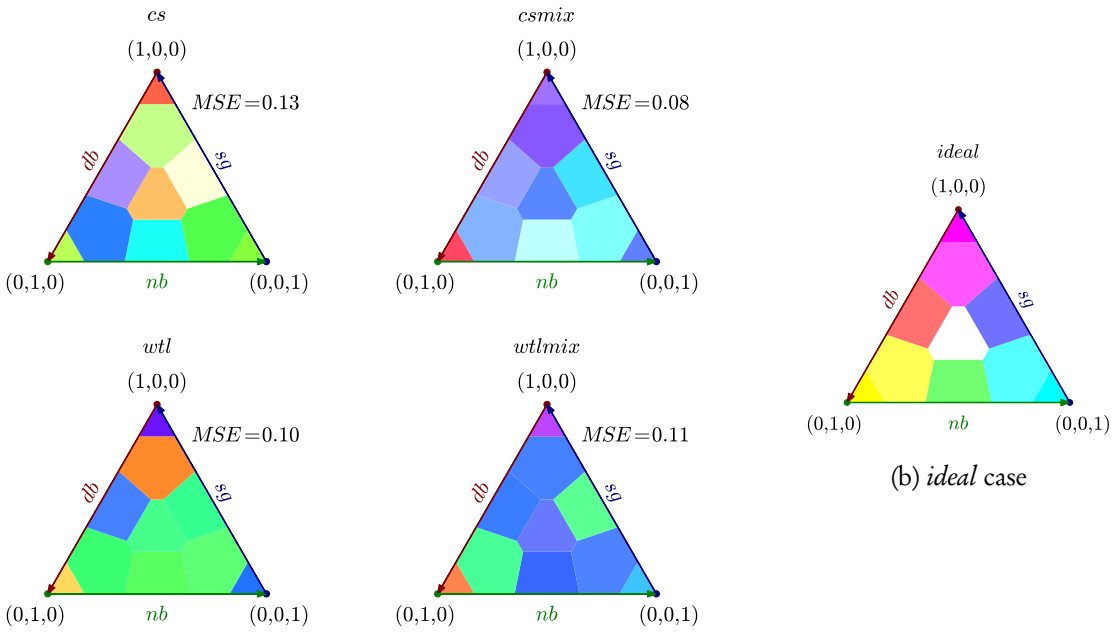


Figure 5.6: Inter-coordinate perceptual convincingness for each method. The shading of each voronoi cell corresponds to the average preference of its respective synthesis coordinate, given the specific mixing technique.

differences between grains. From Figure 5.6, we can see that *csmix* appeared to have the least variance of preference between coordinates, possibly indicating a more reliable mix throughout the coordinate space.



(a) Mixing techniques

(b) *ideal* case

Figure 5.7: Source relevance. Each source recording, *db*, *nb*, and *sg* are colored red, green, and blue, respectively. The color of the voronoi cell corresponding to each synthesis point is an additive blend of the source colors given their source similarity vector, *sim*. The *ideal* plot shows the ideal distribution, in which each cell is colored according to its trilinear coordinates.

5.6.3 Source relevance

Figure 5.7 demonstrates how well the different synthesis algorithms morphed between the properties of their source recordings. The source recordings, *db*, *nb*, and *sg* are colored red, green, and blue, respectively. Based on the distribution of similarity ratings, the colors are additively blended. The final, *ideal* plot, shows the optimal case, where each coordinate has a similarity vector proportional to the distance from each recording.

For each coordinate and synthesis method, the similarity vector is calculated as in Equation 5.13, where S is the set of all three source clips for the given coordinate. The

normalized similarity vector is then defined by

$$sim_i := \frac{pref(s_i)}{\sum_{j=1}^{3} pref(s_j)} \tag{5.14}$$

The mean squared error (MSE) of the similarity vectors from the *ideal* case is also reported for each method.

Very few of the methods seem to provide a particularly smooth blending between the sources. The algorithm that seems to perform best, *csmix*, seems to blend well except for the *db* / *nb* vertex. This may be due to the content of these two recordings–perhaps the features in *nb* are not particularly discernible when directly mixed with those of *db*.

5.7 Conclusions

This chapter has presented an evaluation of four synthesis algorithms for parametrically morphing between two or more environmental audio recordings, two of which use basic concatenative synthesis, and two of which use wavelet tree learning. While pairwise comparisons for perceptual convincingness seem to indicate that listeners prefer a multi-stream concatenative synthesis where WTL is used to re-synthesize sound grains, comparing the synthesized sounds to their original sources seems to indicate that few of the algorithms actually mix the sources in a predictable manner. Closer study will be needed to understand the relationship between what users consider perceptually convincing and what properties of the original sources are actually representative in the mix. Specifically, it may be interesting to do pairwise similarity comparisons between portions of the source clips themselves to create a baseline for how similarity is distributed throughout the coordinate space. Doubtless many other algorithms and versions of those listed above can be applied to the problem of audio texture morphing, but the evaluation platform and associated metrics provide performance baselines and a unified framework for developing future techniques.

Chapter 6

CONCLUSIONS AND FUTURE WORK

This dissertation has provided the following contributions toward the goal of computational soundscape analysis and synthesis:

1. An open source tool, *Sirens*, for the segmentation and comparison of continuous environmental field recordings, allowing users to automatically decompose long recordings into individual sonic events and then compare how similar they are to each other in terms of acoustic content.

2. A unified graph-based framework that allows for the indexing of sound events and associated semantic tags, making use of both acoustic similarity from *Sirens* and WordNet-derived metrics of semantic similarity to allow for both content-based retrieval (query by example), text-based retrieval, and automatic annotation of sounds.

3. A review of existing interactive sound mapping projects, evaluating their various features, content, and scope.

4. The evaluation of several new sound texture synthesis techniques for parametrically morphing between multiple source sound textures for use in an interactive sound map.

These tools are only the very beginning of potential applications of multimedia information retrieval and synthesis to soundscape studies. In addition to the individual conclusions provided at the end of each chapter, several important future directions exist.

To begin, the retrieval framework outlined in Chapter 3 de-contextualizes sound events or objects from their source recordings. However, field recordings have a natural geographic component, especially in interactive sound maps. Since we have a combination of acoustic, semantic, and geographic information, then, this retrieval framework could be extended to the concept of *geographic retrieval*—that is, the study of acoustic or semantic content's distribution in space. One could imagine a system where, by querying with a particular term, the retrieval network could be used to construct a two-dimensional heatmap or contour plot of its *spatial* likelihood surface, as each sound object is associated with a latitude and longitude pair. One challenge of this extension would be in terms of evaluation: how do we determine the "accuracy" of these heatmaps when any assumed ground truth of spatial activity is extremely transient? Perhaps, then, the study could be specified in terms of analysis of the recordist's recording session rather than the continuous activity of the space itself.

Additionally, several possible extensions exist for the re-sonification of soundscapes from multiple source recordings. Chapter 5 explores a few possibilities, but it is clear from human subject evaluation that much work still needs to be done to develop techniques that both sound perceptually convincing and accurately blend between the sources. One possibility is to consider a more ecologically relevant synthesis model that better reflects how humans perceive soundscapes. For example, a model that uses some form of figure/ground separation to independently re-sonify extracted sonic "events" from their background din, which is then blended, may both reduce the number of artifacts introduced by the morphing technique and better reflect how any intermediate point may have actually sounded to a hypothetical listener.

An important caveat, however, is that any technique that relies heavily on source separation and event segmentation introduces artifacts that do not reflect the actual immediate experiences of a listener, as discussed in Chapter 1. Perhaps, then, it is better to

approach these goals from the standpoint of a composition in which there is no attempt to hide the presence of the underlying process. From the standpoint of human-computer interaction, this can be seen as emphasizing *transparency* by properly communicating to the listener what can and cannot be assumed using these tools.

REFERENCES

[1] R. M. Schafer, *The Soundscape: Our Sonic Environment and the Tuning of the World*. Rochester, VT: Destiny Books, 1977.

[2] B. Truax, *Acoustic Communication*. Norwood, NJ: Ablex Publishing, 1984.

[3] B. Truax, "Genres and techniques of soundscape composition as developed at simon fraser university," *Organised sound*, vol. 7, no. 01, pp. 5–14, 2002.

[4] B. Krause, "The niche hypothesis: a virtual symphony of animal sounds, the origins of musical expression and the health of habitats," *The Soundscape Newsletter*, vol. 6, pp. 4–6, 1993.

[5] S. Feld and K. H. Basso, *Senses of place*. School of American Research Press; Distributed by the University of Washington Press, 1996.

[6] S. Feld, *Sound and Sentiment: Birds, Weeping, Poetics, and Song in Kaluli Expression, with a New Introduction by the Author*. Duke University Press, 2012.

[7] B. Blesser and L.-R. Salter, *Spaces Speak, Are You Listening?: Experiencing Aural Architecture*. Cambridge, MA: The MIT Press, 2007.

[8] B. Krause, *The great animal orchestra: finding the origins of music in the world's wild places*. New York: Little, Brown and Company, 2012.

[9] G. Hempton and J. Grossmann, *One Square Inch of Silence: One Man's Search for Natural Silence in a Noisy World*. New York: Simon and Schuster, 2009.

[10] J. Foote, "An overview of audio information retrieval," *Multimedia Systems*, vol. 7, pp. 2–10, Jan. 1999.

[11] G. Guo and S. Z. Li, "Content-based audio classification and retrieval by support vector machines," *IEEE Transactions on Neural Networks*, vol. 14, no. 1, pp. 209–215, 2003.

[12] G. Chechik, E. Ie, M. Rehn, S. Bengio, and D. Lyon, "Large-scale content-based audio retrievals from text queries," in *Proceedings of the 1st International ACM Conference on Multimedia Information Retrieval (MIR 2008)*, (Vancouver, B.C.), pp. 105–112, August 2008.

[13] M. D. Plumbley, S. A. Abdallah, J. P. Bello, M. E. Davies, G. Monti, and M. B. Sandler, "Automatic music transcription and audio source separation," *Cybernetics and Systems*, vol. 33, no. 6, pp. 603–627, 2002.

[14] P. Comon and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*. Oxford: Academic Press, 2010.

[15] J.-J. Aucouturier, "Representing musical genre: A state of the art," *Journal of New Music Research*, vol. 32, no. 1, pp. 83–93.

[16] G. Tzanetakis and P. R. Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, 2002.

[17] Z. Fu, G. Lu, K. M. Ting, and D. Zhang, "A survey of audio-based music classification and annotation," *IEEE Transactions on Multimedia*, vol. 13, no. 2, pp. 303–319, 2011.

[18] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, pp. 467–476, Feb. 2008.

[19] G. Wichern, J. Xue, H. Thornburg, B. Mechtley, and A. Spanias, "Segmentation, indexing, and retrieval for environmental and natural sounds," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 3, pp. 688–707, 2010.

[20] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. New York: Prentice Hall, 1978.

[21] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," in *Proceedings of the IEEE*, vol. 77, pp. 257–286, February 1989.

[22] B. Truax, *Handbook for Acoustic Ecology*. No. 5 in Music of the Environment Series, World Soundscape Project, Cambridge Street Publishing, 1978.

[23] E. Thompson, *The Soundscape of Modernity: Architectural Acoustics and the Culture of Listening in America, 1900-1933*. Cambridge, MA: The MIT Press, 2004.

[24] A. S. Bregman, *Auditory Scene Analysis: The Perceptual Organization of Sound*. Cambridge, MA: The MIT Press, 1994.

[25] J. G. Neuhoff, ed., *Ecological Psychoacoustics*. San Diego, CA: Academic Press, 2004.

[26] W. W. Gaver, "How do we hear in the world? explorations in ecological acoustics," *Ecological Psychology*, vol. 5, no. 4, pp. 285–313, 1993.

[27] W. W. Gaver, "What in the world do we hear?: An ecological approach to auditory event perception," *Ecological Psychology*, vol. 5, no. 1, pp. 1–29, 1993.

[28] P. Schaeffer, *Traité des objets musicaux*. Paris: Le Seuil, 1966.

[29] M. Chion, *Guide To Sound Objects: Pierre Schaeffer and Musical Research*. Buchet Chastal, 1994.

[30] J. Cage, "Listen." Documentary by M. Sebestik. ARTE France Développement, 2003.

[31] P. Oliveros, *Deep Listening: A Composer's Sound Practice*. Bloomington, IN: iUniverse, 2005.

[32] S. Suzuki, *Zen Mind, Beginner's Mind*. New York, NY: Weatherhill, 1973.

[33] A. Watts, *In My Own Way: An Autobiography*. New York, NY: Pantheon Books, 1972.

[34] C. Mydlarz, "Sound Around You." `http://soundaroundyou.com/`, 2009.

[35] G. Wichern, H. Thornburg, B. Mechtley, and A. Spanias, "Robust multi-feature segmentation and indexing for natural sound environments," in *Proceedings of the Fifth International Workshop on Content-Based Multimedia Indexing (CBMI '07)*, (Bordeaux, France), pp. 69–76, 2007.

[36] V. Bush, "As we may think," *Atlantic Monthly*, July 1945.

[37] D. P. W. Ellis and K. Lee, "Minimal-impact audio-based personal archives," in *Proceedings of the First ACM Workshop on Continuous Archival and Retrieval of Personal Experiences (CARPE '04)*, (New York, NY), pp. 39–47, 2004.

[38] C. Cannam, "The Vamp audio analysis plugin api: A programmer's guide." `http://www.vamp-plugins.org/guide.pdf`.

[39] C. Cannam, C. Landone, and M. Sandler, "Sonic visualiser: An open source application for viewing, analysing, and annotating music audio files," in *Proceedings of the ACM Multimedia 2010 International Conference*, (Firenze, Italy), pp. 1467–1468, October 2010.

[40] The Audacity development team, "Audacity: Free audio editor recorder." `http://audacity.soureforge.net/`, 2011.

[41] J. Bullock, "Libxtract." `http://libxtract.sourceforge.net/`, 2007.

[42] G. Tzanetakis and P. R. Cook, "A framework for audio analysis based on classification and temporal segmentation," in *Proceedings of the 25th EUROMICRO Conference*, (Milan, Italy), pp. 61–67, 1999.

[43] P. Cano, M. Koppenberger, S. L. Groux, P. Herrera, J. Ricard, and N. Wack, "Knowledge and content-based audio retrieval using WordNet," in *Proceedings of the 1st International Conference on E-business and Telecommunication Networks (ICETE)*, (Setúbal, Portugal), pp. 301–308, 2004.

[44] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet, "Audio information retrieval using semantic similarity," in *Proceedings of the 2007 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '07)*, vol. 2, pp. 725–728, 2007.

[45] J. Kogan and D. Margoliash, "Automated recognition of bird song elements from continuous recordings using dynamic time warping and hidden markov models: A comparative study," *The Journal of the Acoustical Society of America*, vol. 103, p. 2185, 1998.

[46] S. Parsons and G. Jones, "Acoustic identification of twelve species of echolocating bat by discriminant function analysis and artificial neural networks," *Journal of Experimental Biology*, vol. 203, no. 17, pp. 2641–2656, 2000.

[47] J. Pierce, "Introduction to pitch perception," in *Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics* (P. R. Cook, ed.), ch. 5, 2001.

[48] H. Fletcher and W. Munson, "Loudness, its definition, measurement and calculation," *Journal of the Acoustical Society of America*, vol. 5, pp. 82–108, 1933.

[49] J. O. Smith III and J. S. Abel, "Bark and ERB bilinear transforms," *IEEE Transactions of Speech and Audio Processing*, vol. 7, no. 697-708, 1999.

[50] S. Rickard and M. Fallon, "The GINI index of speech," in *Proceedings of the 38th Conference on Information Science and Systems (CISS '04)*, 2004.

[51] H. Thornburg and R. J. Leistikow, "A new probabilistic spectral pitch estimator: Exact and MCMC-approximate strategies," *Lecture Notes in Computer Science*, no. 3310, 2005.

[52] P. R. Cook, "Pitch, periodicity, and noise in the voice," in *Music, Cognition, and Computerized Sound: An Introduction to Psychoacoustics* (P. R. Cook, ed.), ch. 16, 2001.

[53] J. L. Goldstein, "An optimum processor theory for the central formation of the pitch of complex tones," *Journal of the Acoustical Society of America*, vol. 54, pp. 1496–1516, 1973.

[54] G. Tzanetakis and P. R. Cook, "Multifeature audio segmentation for browsing and annotation," in *Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, (New Paltz, NY), 1999.

[55] A. Sheh and D. P. W. Ellis, "Chord segmentation and recognition using EM-trained hidden Markov models," in *Proceedings of the 4th International Conference on Music Information Retrieval (ISMIR)*, (Washington, D.C.), pp. 183–189, 2003.

[56] J. Foote, "Automatic audio segmentation using a measure of audio novelty," in *Proceedings of the 2004 IEEE International Conference on Multimedia and Expo (ICME 2004)*, (Los Alamitos, CA), pp. 452–455, 2000.

[57] V. Pavlovic, J. M. Rehg, and T. Cham, "A dynamic Bayesian network approach to tracking learned switching dynamic models," in *Proceedings of the International Workshop on Hybrid Systems*, (Pittsburgh, PA), 2000.

[58] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2 ed., 2007.

[59] G. Wichern, *Robust Segmentation and Retrieval of Environmental Sounds*. Doctoral thesis, Arizona State University, 2010.

[60] G. Wichern, J. Xue, H. Thornburg, and A. Spanias, "Distortion-aware query by example for environmental sounds," in *Proceedings of the 2007 IEEE Workshop on Applications of Signal Processing to Audio Acoustics (WASPAA '07)*, (New Paltz, NY), pp. 335–338, 2007.

[61] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *AAAI Workshop on Knowledge Discovery in Databases (KDD)*, vol. 10, pp. 359–370, Seattle, WA, 1994.

[62] B. Whitman and D. P. W. Ellis, "Automatic record reviews," in *Proceedings of the Fourth International Conference on Music Information Retrival (ISMIR '04)*, (Barcelona, Spain), pp. 470–477, 2004.

[63] S. Kim, S. Narayana, and S. Sundaram, "Acoustic topic model for audio information retrieval," in *Proceedings of the 2009 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA '09)*, (New Paltz, NY), pp. 37–40, 2009.

[64] T. Li and M. Ogihara, "Detecting emotion in music," in *Proceedings of the 4th International Symposium on Music Information Retrieval (ISMIR)*, (Baltimore, MD), pp. 239–240, 2003.

[65] S. Essid, G. Richard, and B. David, "Inferring efficient hierarchical taxonomies for music information retrieval tasks: Application to musical instruments," in *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR)*, (London), pp. 324–328, 2005.

[66] P. Herrera-Boyer, G. Peeters, and S. Dubnov, "Automatic classification of musical instrument sounds," *Journal of New Music Research*, vol. 32, no. 1, pp. 3–21, 2003.

[67] L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet, "Combining feature kernels for semantic music retrieval," in *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR)*, (Philadelphia, PA), pp. 614–619, 2008.

[68] G. Wichern, B. Mechtley, A. Fink, H. Thornburg, and A. Spanias, "An ontological framework for retrieving environmental sounds using semantics and acoustic content," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2010, pp. 1–11, 2010.

[69] B. Mechtley, G. Wichern, H. Thornburg, and A. Spanias, "Combining semantic, social, and acoustic similarity for retrieval of environmental sounds," in *Proceedings of the 2010 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2010)*, (Dallas, TX), pp. 2402–2405, 2010.

[70] Universitat Pompeu Fabra Music Technology Group, "The freesound project." `http://www.freesound.org/`.

[71] M. Goto and K. Hirata, "Recent studies on music information processing," *Acoustical Science and Technology*, vol. 25, no. 6, 2004.

[72] O. C. M. Sordo, C. Laurier, "Annotating music collections: How content-based similarity helps to propagate labels," in *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR)*, (Vienna, Austria), 2007.

[73] P. Cano and M. Koppenberger, "Automatic sound annotation," in *IEEE Workshop on Machine Learning for Signal Processing*, pp. 391–400, 2004.

[74] P. Cano, M. Koppenberger, S. Le Groux, J. Ricard, P. Herrera, and N. Wack, "Nearest-neighbor generic sound classification with a WordNet-based taxonomy," in *Proceedings of the 116th AES Convention*, (Berlin, Germany), 2004.

[75] E. Martínez, O. Celma, M. Sordo, B. de Jong, and X. Serra, "Extending the folksonomies of freesound.org using content-based audio analysis," in *Proceedings of the 6th Sound and Music Computing Conference*, (Porto, Portugal), pp. 23–25, 2009.

[76] M. Slaney, "Semantic audio retrieval," in *Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, vol. 4, (Orlando, FL), pp. 4108–4111, 2002.

[77] F. Crestani, "Application of spreading activation techniques in information retrieval," *Artificial Intelligence Review*, vol. 11, no. 6, pp. 453–482, 1997.

[78] G. Wichern, H. Thornburg, and A. Spanias, "Unifying semantic and content-based approaches for retrieval of environmental sounds," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, (New Paltz, NY), pp. 13–16, 2009.

[79] T. Pedersen, S. Patwardhan, and J. Michelizzi, "Wordnet:: Similarity: measuring the relatedness of concepts," in *Proceedings of the Nineteenth National Conference on Artificital Intelligence (AAAI-04)*, (San Jose, CA), pp. 1024–1025, 2004.

[80] B. H. Huang and L. R. Rabiner, "A probabilistic distance measure for hidden Markov models," *AT&T Technical Journal*, vol. 64, no. 2, pp. 1251–1270, 1985.

[81] A. Budanitsky and G. Hirst, "Evaluating wordnet-based measures of lexical semantic relatedness," *Computational Linguistics*, vol. 32, no. 1, pp. 13–47, 2006.

[82] "Semantic similarity based on corpus statistics and lexical taxonomy," in *Proceedings of International Conference Research on Computational Linguistics (ROCLING X)*, (Taiwan), 1997.

[83] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll, "Finding predominant word senses in untagged text," in *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, (Barcelona, Spain), pp. 280–287, 2004.

[84] R. Mandala, T. Tokunaga, and H. Tanaka, "The use of WordNet in information retrieval," in *Proceedings of the Workshop on Usage of WordNet in Natural Language Processing Systems*, (Montréal, QC), pp. 31–37, 1998.

[85] B. Mechtley, P. Cook, and A. Spanias, "Shortest path techniques for annotation and retrieval of environmental sounds," in *Proceedings of the 13th International Conference on Music Information Retrieval (ISMIR)*, (Porto, Portugal), 2012.

[86] B. Mechtley, P. Cook, and A. Spanias, "Sound mapping on the web: Current solutions and future directions," in *Proceedings of The Symposium on Acoustic Ecology*, (Chatham, United Kingdom), 2013.

[87] OpenStreetMap contributors, "OpenStreetMap." `http://www.openstreetmap.org/`, 2011.

[88] SoundCloud, Ltd., "SoundCloud." `http://soundcloud.com`, 2007.

[89] O. Bélanger, "pyo." `https://code.google.com/p/pyo/`, 2007.

[90] The Center For New Music and Audio Technology (CNMAT), UC Berkeley, "Open sound control." `http://opensoundcontrol.org/`, 2002.

[91] M. Puckette, "Puredata." `http://puredata.info`, 1997.

[92] Cycling 74, "Max/MSP 6.1.2." `http://cycling74.com/products/max`, 2013.

[93] G. Wang, S. Salazar, P. Cook, R. Fiebrink, A. Misra, A. Lazier, P. Davidson, and D. Trueman, "Chuck." `http://chuck.cs.princeton.edu/`, 2002.

[94] S. Dubnov, Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman, "Synthesizing sound textures through wavelet tree learning," *IEEE Computer Graphics and Applications*, vol. 22, no. 4, pp. 38–48, 2002.

[95] D. Schwarz, "State of the Art in Sound Texture Synthesis," in *Proceedings of the 14th International Conference on Digital Audio Effects (DAFx-11)*, (Paris, France), pp. 221–231, 2011.

[96] Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman, "Texture mixing and texture movie synthesis using statistical learning," *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 2, pp. 120–135, 2001.

[97] M. Fröjd and A. Horner, "Sound Texture Synthesis Using an Overlap âĂŞ Add / Granular Synthesis Approach," *Journal of the Audio Engineering Society*, vol. 57, no. 1/2, pp. 29–37, 2009.

[98] J. R. Parker and B. Behm, "Creating audio textures by example: tiling and stitching," in *Proceedings of the 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 4, (Montréal, QC), pp. 317–320, IEEE, 2004.

[99] B. Truax, "Real-time granular synthesis with a digital signal processor," *Computer Music Journal*, vol. 12, no. 2, pp. 14–26, 1988.

[100] D. Keller and B. Truax, "Ecologically-based granular synthesis," in *Proceedings of the International Computer Music Conference*, pp. 117–120, 1998.

[101] D. Schwarz, *Data-Driven Concatenative Sound Synthesis*. Doctoral thesis, Institut de Recherche et Coordination Acoustique/Musique, 2004.

[102] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proceedings of SIGGRAPH 2000*, (New York, NY), pp. 479–488, 2000.

[103] Z. Bar-Joseph, "Statistical learning of multi-dimensional textures," master's thesis, The Hebrew University of Jerusalem, 1999.

[104] A. Misra, *TAPESTREA: Techniques and Paradigms for Expressive Synthesis, Transformation, and Re-composition of Environmental Audio*. Doctoral thesis, Princeton University, 2009.

[105] D. O'Regan and A. Kokaram, "Multi-resolution sound texture synthesis using the dual-tree complex wavelet transform," in *Proceedings of the 2007 European Signal Processing Conference (EUSIPCO 2007)*, (Poznand, Poland), pp. 350–354, 2007.

APPENDIX A

LIST OF SOUND MAPS

Title	Authors	Year	Features	Media	Content	Metadata	Host
1. 12 Gates to the City Acoustic Map	Jonathan Prior	2009	point-and-click		field recordings	descriptions	Custom
Location: http://12gatestothecity.com/acoustic-map/							
2. 12 Stations	Softday	2011	gps traces point-and-click		field recordings	descriptions	Custom
Location: http://www.softday.ie/wld2011/							
3. Accents and dialects	The British Library	2011	point-and-click		historical data speech only	categories descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Accents-and-Dialects							
4. An Interactive Soundmap of Wales	Community Music Wales	2013	point-and-click		community field recordings	descriptions	Soundcloud
Location: http://www.mapsain.org.uk/							
5. Audioboo	Audioboo Ltd.	2011	mobile	images	commercial community	categories descriptions tags	Custom
Location: http://www.audioboo.com/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
6. AudioBus B61	Andrea Callard and Fred Krughoff	2006	gps traces point-and-click	images	field recordings		Custom
Location: http://www.andreacallard.com/audiobus/							
7. Berlin Wall of Sound	Netaudio Berlin	2009	point-and-click		field recordings	descriptions	Custom
Location: http://www.netaudioberlin.de/berlin-wall-of-sound/							
8. Bolognoise	Ricardo Dance, Federico Mascagni, and Salvatore Arangio	2012	autoplay point-and-click		community field recordings	categories descriptions	Custom
Location: http://bolognoise.org/							
9. Broadcastr	Electric Literature, LLC	2010					Custom
Location: http://broadcastr.com/							
10. Brussels SoundMap	Harry + LVH	2011	autoplay autoplay next point-and-click sequences		community field recordings	descriptions	Custom
Location: http://www.bna-bbot.be/brusselssoundmap/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
11. Carleton University Soundmap Project (CUSP)	Carleton Sound at Carleton University	2012	point-and-click		field recordings	descriptions	Custom
Location: http://www3.carleton.ca/clubs/carletonsound/soundmap/							
12. Cartophonies	Grégoire Cheikoff, Françoise Acquier, Julien Moisans, Sylvie Laroche, and Gabriel Bérubé	2012	autoplay autoplay next point-and-click sequences		community field recordings	descriptions	Custom
Location: http://www.cartophonies.fr/							
13. Cinco Cidades Soundmap	Alastair Dant, Tom Davis, and David Gunn	2007	autoplay mixing point-and-click		field recordings	descriptions	Custom
Location: http://www.cincocidades.com/							
14. Citygram	Tae-Hong Park	2013			field recordings		Custom
Location: http://citygram.calarts.edu/							
15. Davos Soundscape	Jan Schacher and Marcus Maeder	2007	point-and-click		field recordings	descriptions	Custom
Location: http://www.davosoundscape.ch/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
16. Earth Ear Soundmap	David Dunn, Steve Feld, Tom Lawrence, David Monacchi, Jay Needham, Doug Quin, Jason Reinier, David Rothenberg, Lisa Walker, and Mariolina Zitta		point-and-click		commercial field recordings	descriptions	Custom
Location: http://earthear.com/soundmap.html							
17. escoitar	Berio Molina, Chiu Longina, Horacio González, Juan-Gil Rodrïguez, Julio Gómez, and Carlos Suárez Sánchez	2006	autoplay point-and-click		community field recordings	categories descriptions	Custom
Location: http://www.escoitar.org/							
18. Favorite Chicago Sounds	Lou Mallozzi, Teresa Gale, Sarah Lu, and Andrew Neher	2008	point-and-click		field recordings	categories descriptions	Custom
Location: http://www.favoritechicagosounds.com/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
19. Favourite Sounds	Peter Cusack, Nigel Currie, and Kirsten Edwards	2011	autoplay autoplay next mixing point-and-click sequences		field recordings	categories descriptions tags	Custom
Location: http://favouritesounds.org/							
20. Firenze Sound Map	Antonella Chicory	2009	point-and-click	images	field recordings	categories	Custom
Location: http://www.firenzesoundmap.org/							
21. Folk Songs for the Five Points	Alastair Dant, Tom Davis, and David Gunn		point-and-click		field recordings		Custom
Location: http://www.tenement.org/folksongs/client/							
22. GeoGraffiti	Blingpost, LLC.	2011					Custom
Location: http://www.geograffiti.com/							
23. Gordon Soundscape	Pete Stollery and Phil Marston	2005	point-and-click		field recordings	descriptions	Custom
Location: http://homepages.abdn.ac.uk/wae006/gordonsoundscape.co.uk/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
24. Hear and There	Joey Rozier, Karrie Karahalios, and Judith Donath	1999			field record-ings		Custom
Location: http://smg.media.mit.edu/projects/HearAndThere/							
25. Inukjuak Sound Map	Nimalan Yoganathan and Maxwell Stein	2010	autoplay point-and-click		field record-ings	categories descriptions	Custom
Location: http://www.inukjuaksoundmap.com/							
26. Invisible Valley Sound Map	Miguel Isaza	2011	point-and-click		field record-ings	descriptions	Custom
Location: http://invisiblevalley.com/map/							
27. Ipswich Sound Map	Enquiring Ear	2013	point-and-click		field record-ings	descriptions	Custom
Location: http://www.suffolkbirds.co.uk/sounds/soundmap/index.html							
28. Jewish survivors of the Holocaust	The British Library	2011	point-and-click		historical data speech only	descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Jewish-Holocaust-Survivors							

Title	Authors	Year	Features	Media	Content	Metadata	Host
29. k146: cartographie sonore autour du Taurion	Cédric Peyronnet	2005	point-and-click		field recordings	descriptions	Custom
Location: http://www.k146.org/category/cartes-maps/							
30. Klaus Wachsmann Uganda recordings	The British Library	2011	point-and-click		music only	descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Wachsmann							
31. La Ville S'onore: Carte sonore subjective	Gwladys Déprez and Sabine Petit	2009	point-and-click		field recordings	descriptions	Custom
Location: http://lavillesonore.fr/blog/?p=174							
32. Listen to Africa	H. Williams and R. Summer	2009	point-and-click	images videos	field recordings	categories descriptions tags	Custom
Location: http://www.listentoafrica.com/map/							
33. Listening to the Deep Ocean Environment	Michel André, Mike van der Schaar, Serge Zaugg, Ludwig Houégnigan, Antionio M. Sánchez, and Joan Vicent Castell	2011	live streaming point-and-click	spectrogram	field recordings	descriptions	Custom
Location: http://www.listentothedeep.net							

Title	Authors	Year	Features	Media	Content	Metadata	Host
34. Locustream Map	Locus Sonus	2006	live streaming point-and-click	images	community field recordings	descriptions	Custom
Location: http://www.locusonus.org/soundmap/							
35. London Sound Survey	Ian Rawes	2011	point-and-click	additional map layers images	field recordings historical data	categories descriptions	Custom
Location: http://www.soundsurvey.org.uk/							
36. Macaulay Library	The Cornell Lab of Ornithology	2011			commercial field recordings	categories descriptions	Custom
Location: http://macaulaylibrary.org/							
37. Madrid Soundscape	Berio Molina, Chiu Longina, Horacio González, Juan-Gil Rodrïguez, Julio Gómez, and Carlos Suárez Sánchez	2008	autoplay point-and-click		community field recordings	categories descriptions	Custom
Location: http://www.madridsoundscape.org/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
38. Make Sound Here	James Saunders	2013	point-and-click		field recordings	descriptions	Audioboo
Location: http://makesoundhere.com/wp/							
39. Mapa Sonoro das Aldeias do Concelho de São Pedro do Sul	Luis Costa and Manuela Barile	2013	point-and-click		field recordings	descriptions	Custom
Location: http://www.aldeias-sonoras.org/mapas-de-sons/s-pedro-do-sul							
40. Mapa Sonoro de Guayaquil	Juan José Ripalda, Fernando Alava, and Laura San Andrés	2012	point-and-click	images	community field recordings	descriptions	Custom
Location: http://mapasonorodeguayaquil.com							
41. mapaSONOU	Juanjo Palacios	2009	point-and-click		community field recordings	descriptions	Custom
Location: http://mapasonoru.com/							
42. Memoryscape Audio walks	Toby Butler	2005	point-and-click		field recordings	descriptions	Custom
Location: http://www.memoryscape.org.uk							

Title	Authors	Year	Features	Media	Content	Metadata	Host
43. Mississauga Sound Map	Don Sinclair and Hector Centeno	2008	point-and-click		field recordings	descriptions	Custom
Location: http://www.yorku.ca/caseaces/soundmap/							
44. MoMA Studio Sound Map	MoMA Studio	2013	autoplay point-and-click		community field recordings		Soundcloud
Location: http://www.moma.org/moma_studio/soundmap							
45. Montréal Sound Map	Maxwell Stein and Julian Stein	2008	autoplay autoplay next point-and-click		community field recordings	categories descriptions tags	Custom
Location: http://www.montrealsoundmap.com/							
46. Morvan Auxois	Cédric Peyronnet	2008	point-and-click		field recordings	descriptions	Custom
Location: http://morvan-auxois.ingeos.org/							
47. Music from India	The British Library	2011	point-and-click		historical data music only	descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Music-From-India							

Title	Authors	Year	Features	Media	Content	Metadata	Host
48. NOAA Underwater Sound Field Working Group	NOAA	2011					Custom
Location: http://www.st.nmfs.noaa.gov/cetsound/sound.html							
49. Open Sound New Orleans	Heather Booth and Jacob Brancasi	2009	point-and-click	images	community	categories descriptions tags	Custom
Location: http://www.opensoundneworleans.com/							
50. Orcasound.net Hydrophone Network	Salish Sea Hydrophone Network	2011	live streaming point-and-click		field recordings		Custom
Location: http://orcasound.net/							
51. Paris Sound Map	Des Coulam	2011	autoplay point-and-click		field recordings	descriptions	Soundcloud
Location: http://soundlandscapes.eu/paris-sound-map							
52. Passages Couverts Sound Map	Des Coulam	2011	autoplay point-and-click		field recordings	descriptions	Soundcloud
Location: http://soundlandscapes.eu/passages-couverts-sound-map							

Title	Authors	Year	Features	Media	Content	Metadata	Host
53. Pod Mostom Project	Alena Kislitsing and Vadim Nazarov		autoplay point-and-click	images	field recordings		Custom
Location: http://www.pod-mostom-project.narod.ru/							
54. radio aporee :: maps	Udo Noll	2006	autoplay autoplay next gps traces mixing mobile point-and-click sequences		community field recordings platform	categories descriptions	Custom
Location: http://www.aporee.org/maps/							
55. Save Our Sounds	BBC World Service	2011					Custom
Location: http://www.bbc.co.uk/worldservice/specialreports/saveoursounds.shtml							
56. SeoulSoundMap	Sound@Media	2011	autoplay point-and-click		community field recordings	descriptions	Custom
Location: http://som.saii.or.kr/campaign							

Title	Authors	Year	Features	Media	Content	Metadata	Host
57. Sergiev Posad Sounds	Vladimir Kryutchev	2012	point-and-click		field recordings	categories descriptions	Custom
Location: http://www.oontz.ru/karta/							
58. Soa-te a ISTo	Rádio Zero	2008	point-and-click		field recordings	tags	Custom
Location: http://soa-te.radiozero.pt/							
59. Soinu Mapa	Xabier Erkizia, Iñigo Telletxea, Txesus Garate, Myriam Ayçaguer, Stéphane Garin, Mikel R. Nieto, Aintzane Erkizia, Oier Iruretagoiena, Xavier Balderas, and Enrike Hurtado	2005	autoplay point-and-click		field recordings	categories descriptions tags	Custom
Location: http://www.soinumapa.net/							
60. Sonic Explorers Sound Map	Sonic Explorers	2012	autoplay point-and-click	images	field recordings	categories descriptions	Custom
Location: http://www.sonicexplorers.org/sound-map.html							

Title	Authors	Year	Features	Media	Content	Metadata	Host
61. Sonic Sidewalks	Softday	2010	gps traces point-and-click		field recordings	descriptions	Custom
Location: http://www.softday.ie/sonicsidewalks/							
62. Sons de Barcelona	Music Technology Group of Pampeu Fabra University	2008	point-and-click		field recordings	descriptions tags	Freesound
Location: http://barcelona.freesound.org/							
63. Sound Archives of the CNRS - Musée de l'Homme	Centre de Recherche en Ethnomusicologie	2013	point-and-click		music only	categories descriptions spectrogram tags visualisations	Custom
Location: http://archives.crem-cnrs.fr/							
64. Sound Around You	Charlie Mydlarz	2009	mobile point-and-click		community field recordings	descriptions	Custom
Location: http://soundaroundyou.com/							
65. Sound Maps of London Road	London Road Sounds	2010	autoplay point-and-click		field recordings		Custom
Location: http://www.londonrdsounds.co.uk/map/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
66. Sound Tourism	Trevor Cox	2011	point-and-click	images videos	field recordings	categories descriptions tags	Custom
Location: http://www.sonicwonders.org/							
67. Sound-Seeker	Andrea Polli and Sha Sha Feng	2004	point-and-click		community field recordings	descriptions	Custom
Location: http://www.soundseeker.org/							
68. Soundcities	Stanza	2000	autoplay point-and-click		community field recordings	categories descriptions	Custom
Location: http://www.soundcities.com/							
69. soundingD SoundMap	Robin Minard ad Hennig Ludger		autoplay mixing point-and-click		field recordings	descriptions	Custom
Location: http://map.sounding-d.de/map							
70. Soundmap of Cologne	Marco Medkour	2013	point-and-click		field recordings	categories descriptions tags	Custom
Location: http://soundmap-cologne.de							

Title	Authors	Year	Features	Media	Content	Metadata	Host
71. Soundmap of London canals and minor rivers	Ian Rawes	2012	point-and-click	additional map layers images	field recordings historical data	descriptions	Custom
Location: http://www.soundsurvey.org.uk/index.php/survey/waterways/							
72. Soundmap of the Paiva River	Luis Costa and Manuela Barile	2013	point-and-click		field recordings	descriptions	Custom
Location: http://www.aldeias-sonoras.org							
73. Soundscapes	The British Library	2011	point-and-click		field recordings	descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Soundscapes							
74. soundtrack	Gokce Kinayoglu	2011	gps traces point-and-click		community field recordings		Custom
Location: http://www.dmrlab.org/402/							
75. SoundTransit	Derek Holzer and and Marc Boon	2004	sequences		field recordings	descriptions	Custom
Location: http://www.turbulence.org/soundtransit/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
76. SoundTrips	SoundTrips	2011	point-and-click		field recordings	descriptions tags	Custom
Location: http://soundtrip.org/							
77. Staten Island Sound Map	Sounds Like Staten Island	2009					Custom
Location: http://www.soundslikestatenisland.com/map/							
78. Street Sounds	The Smalls	2009					Custom
Location: http://www.thesmalls.com/StreetSounds/							
79. Sydney Sidetracks	Australian Broadcasting Corporation	2011	point-and-click	images videos	historical data	descriptions	Custom
Location: http://www.abc.net.au/innovation/sidetracks/							
80. Tactical Sound Garden	M. Shepard, F. Murphy, B. Diesel, A. Flynt, A. Thomas, V. Modi, and A. Krishnamurthy	2007	autoplay mixing point-and-click		field recordings		Custom
Location: http://www.tacticalsoundgarden.net/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
81. Taiwan SoundMap Project	Tsan-Cheng Wu		point-and-click		field recordings		Radio Aporee, Soundcloud
Location: http://www.soundandtaiwan.com/soundmap/main.html							
82. Telemeta	Guillaume Pellerin, Riccardo Zaccarelli, and Olivier Guilyardi	2013	platform point-and-click			categories descriptions spectrogram tags	Custom
Location: http://telemeta.org/							
83. Thames estuary sound map recordings	Ian Rawes	2010	point-and-click	additional map layers images	field recordings historical data	descriptions	Custom
Location: http://www.soundsurvey.org.uk/index.php/survey/estuary/							
84. The Freesound Project	Universitat Pampeu Fabra Music Technology Group	2005	point-and-click	spectrogram	community field recordings platform	descriptions tags	Custom
Location: http://freesound.org/							

Title	Authors	Year	Features	Media	Content	Metadata	Host
85. Toronto Island Sound Map	Don Sinclair, Diego Philips-Shea, and Darren Copeland	2004	autoplay point-and-click		field recordings	descriptions	Custom
Location: http://www.yorku.ca/dws/tism/							
86. Toronto Sound Map	Frank Russio, Tristan Loria, Gabe Nespoli, Max Stein, and Chris Lachine	2012	point-and-click sequences	images	field recordings	categories descriptions	Custom
Location: http://torontosoundmap.com							
87. Traditional music in England	The British Library	2011	point-and-click		historical data music only	descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Traditional-Music-In-England							
88. UK Soundmap	The British Library	2011	point-and-click		community field recordings		Audioboo
Location: http://sounds.bl.uk/Sound-Maps/UK-Soundmap/							
89. Urban Sound Ecology	Urban Sound Ecology	2011	gps traces point-and-click		field recordings	tags	Custom
Location: http://missionspecialist.net/project/urban-sound-ecology							

Title	Authors	Year	Features	Media	Content	Metadata	Host
90. Western Soundscape Archive Sound Maps	Western Soundscape Archive	2011	point-and-click	images	field recordings	categories descriptions	Custom
Location: http://westernsoundscape.org/googleMap.php							
91. Wildlife recordings	The British Library	2011	point-and-click		field recordings	categories descriptions	Custom
Location: http://sounds.bl.uk/Sound-Maps/Wildlife-Recordings							
92. Wisconsin Geotools Project	Janet Silbernagel, Matthew Axler, and Francis Eanes		point-and-click	images video	community field recordings mobile	description	Custom
Location: http://maps.aqua.wisc.edu/geotools/							
93. xeno-canto	Xeno-canto Foundation	2005	point-and-click	heatmap spectrogram	community field recordings	categories descriptions	Custom
Location: http://www.xeno-canto.org/							
94. Your accents	The British Library	2011	point-and-click		community speech only		Audioboo
Location: http://sounds.bl.uk/Sound-Maps/Your-Accents							

Title	Authors	Year	Features	Media	Content	Metadata	Host
95. Écouter Paris	Monica Fantini, Michel Créïs, Irène Berelowitch, and Xavier Baudoin	2011	autoplay point-and-click		field record-ings	categories	Custom
Location: http://www.ecouterparis.net/							

APPENDIX B

SIRENS AND *SOUNDWALKS* ARCHITECTURE

This appendix discusses the various architectural details and decisions made in the software packages described in Chapters 2 and 4. Full API references can be found in the most recent versions of the respective software packages—this appendix aims to provide basic information about the reasoning behind their design.

B.1 *Sirens*

Sirens is made available as an open source software library and is developed with concurrency in mind in order to efficiently extract features before performing segmentation or comparison routines. This appendix will briefly discuss classes in the *Sirens* architecture, split into three categories: 1) feature extraction, 2) segmentation, and 3) comparison. At the time of this writing, *Sirens* is available at `http://github.com/plant/sirens`.

B.1.1 Feature extraction

The base set of features in *Sirens* consists of six low-level perceptually motivated features including `Loudness`, `TemporalSparsity`, `SpectralCentroid`, `SpectralSparsity`, `TransientIndex`, and `Harmonicity`, each of which is implemented as a class that inherits from the class `Feature`. `Feature` can also be instantiated by itself to support using precomputed feature values. Additionally, the class `FeatureSet` acts as a container for several `Feature` objects that need to be calculated or used for segmentation or retrieval. `FeatureSet` supports adding "sample" or "spectral" features, which take in windowed sample values or FFT bin values, respectively. Below is an example of using the six basic features and a `FeatureSet` to perform feature extraction:

```
// 1. Open sound.
Sound* sound = new Sound();
sound->setFrameLength(0.04); // 40ms Hamming windows.
sound->setHopLength(0.02);   // 20ms hops between windows.
sound->open("somesound.wav");

// 2. Output some information about the sound.
cout << "\tDuration: "
    << double(sound->getSampleCount()) / sound->getSampleRate()
    << "s (" << sound->getSampleCount() << " samples)" << endl;

cout << "\tSample rate: " << sound->getSampleRate() << " Hz" << endl;

cout << "\tFrame length: " << sound->getFrameLength()
    << "s (" << sound->getSamplesPerFrame() << " samples)" << endl;

cout << "\tHop length: " << sound->getFrameLength()
    << "s (" << sound->getSamplesPerHop() << " samples)" << endl;

cout << "\tFFT size: " << sound->getFFTSize() << " samples" << endl;

cout << "\tSpectrum size: " << sound->getSpectrumSize()
    << " bins" << endl;

// 3. Initialize features.
// The first frame of TransientIndex is not defined,
// so it doesn't need to be recorded.
int frames = sound->getFrameCount() - 1;
int spectrum_size = sound->getSpectrumSize();
int sample_rate = sound->getSampleRate();
```

```
Loudness* loudness = new Loudness(frames);
TransientIndex* transient_index = new TransientIndex(
    frames, spectrum_size, sample_rate, 30, 15
);
TemporalSparsity* temporal_sparsity = new TemporalSparsity(frames, 49);
SpectralSparsity* spectral_sparsity = new SpectralSparsity(frames);
SpectralCentroid* spectral_centroid = new SpectralCentroid(
    frames, spectrum_size, sample_rate
);

// Harmonicity has several additional parameters that can be tuned.
Harmonicity* harmonicity = new Harmonicity(
    frames, spectrum_size, sample_rate
);
harmonicity->setAbsThreshold(1);
harmonicity->setThreshold(0.1);
harmonicity->setSearchRegionLength(5);
harmonicity->setMaxPeaks(3);
harmonicity->setLPFCoefficient(0.7);

// 4. Initialize a feature set to hold all the features.
FeatureSet* feature_set = new FeatureSet();
feature_set->addSampleFeature(loudness);
feature_set->addSampleFeature(temporal_sparsity);
feature_set->addSpectralFeature(spectral_sparsity);
feature_set->addSpectralFeature(spectral_centroid);
feature_set->addSpectralFeature(transient_index);
feature_set->addSpectralFeature(harmonicity);

// 5. Extract features.
sound->setFeatureSet(feature_set);
```

```
sound->extractFeatures();

feature_set->saveCSV("features.csv");
```

Listing B.1: Example feature extraction

The above example in Listing B.1 first demonstrates the basics of opening a sound file. The `Sound` class provides a wrapper around the open source library *libsndfile*, which allows *Sirens* to handle many different audio formats. Additionally, an instance of `Sound` contain methods to initiate feature extraction once a feature set has been assigned to it. Part "2" of the code listing demonstrates some of the different methods that are available for gathering information about the loaded sound.

In Part 3 of Listing B.1, the feature objects are initialized. Each feature object needs to be told the size of the expected window of samples (or FFT bins, in the case of spectral features), denoted in this example by the variable `frames`. Some features take additional parameters, such as the number of MFCC bins in the case of `TransientIndex` and various parameters for peak picking in the case of `Harmonicity`.

In Part 4, an instance of `FeatureSet` is constructed which will act as a container for the features that have been created. Note that each feature is added using either `FeatureSet::addSampleFeature` or `FeatureSet::addSpectralFeature`, depending on the type of data the feature uses. When `extractFeatures` is finally called on `sound` in line 63, it computes the fast fourier transform using an instance of `FFT` (using the FFT library *fftw*) and concurrently begins computing all features at the same time, joining their respective threads afterward to suspend until all are complete.

`FeatureSet` contains a method `saveCSV` to output the computed feature trajectories as a comma-separated value (CSV) file, and finally, the user must deallocate the memory used for the sound, features, and feature set. This example is contained in *Sirens* as a standalone application in `examples/features.cpp`.

B.1.2 Segmentation

After features have been extracted, as in Listing B.1, the classes `Segmenter` and `SegmentationParameters` can be used to perform the automatic segmentation described in Section 2.2. Every instance of `Feature` contains a variable `parameters`, which is an instance of `SegmentationParameters`. The various parameters that can be set in these objects reflect the various transition variances described in Section 2.2.

Once the segmentation parameters have been set for each feature, a new `Segmenter` is created, the `FeatureSet` is attached to it, and segmentation can be initiated. Below is an example code listing for setting parameters for the `loudness` feature constructed in Listing B.1:

```
loudness->parameters()->alpha = 0.15;
loudness->parameters()->r = 0.0098;
loudness->parameters()->cStayOff = 0.0015;
loudness->parameters()->cTurnOn = 0.085;
loudness->parameters()->cTurnOff = 0.085;
loudness->parameters()->cNewSegment = 0.085;
loudness->parameters()->cStayOn = 0.05;
loudness->parameters()->pLagPlus = 0.75;
loudness->parameters()->pLagMinus = 0.75;
```

Listing B.2: Example segmentation parameters for loudness.

Additionally, below is a code listing for using a subset of features in a feature set for segmentation:

```
// This example requires std::vector for storing segmentation boundaries
    .
#include <vector>
using namespace std;
```

```
/*
    ...
    Sound loading, feature initialization,
    and feature extraction go here.
    ...
*/

// 1. Create a feature set with features whose segmentation parameters
// have been set. This can be the same feature set used for extraction.
FeatureSet feature_set;
feature_set.addSampleFeature(&loudness);
feature_set.addSpectralFeature(&spectral_centroid);
feature_set.addSpectralFeature(&spectral_sparsity);

// 2. Perform segmentation.
// Parameters include prior "on" and "off" parameters.
Segmenter segmenter(0.00000000001, 0.00000000001);
segmenter.setFeatureSet(&feature_set);
segmenter.segment();

// 3. Obtain segmentation boundaries.
vector<vector<int> > segments = segmenter.getSegments();
vector<int> modes = segmenter.getModes();

// 4. Output segments as CSV (index, start sample, end sample)
for (int i = 0; i < segments.size(); i++) {
    cout << i << "," << segments[i][0] << "," << segments[i][1] << ","
        << segments[i][0] * sound.getSamplesPerHop() << ","
        << segments[i][1] * sound.getSamplesPerHop() << endl;
}
```

```
// 5. Save segments to disk as WAV files.
for (int i = 0; i < segments.size(); i++) {
    sound.saveSegment(
        "segment" + double_to_string(i) + ".wav",
        segments[i][0],                    // Beginning frame index.
        segments[i][1]                     // Ending frame index.
    );
}
```

Listing B.3: Example segmentation with several pre-initialized/extracted features.

In the above example, Listing B.3, once features have been extracted, the same feature set used for extraction or a new feature set, containing only a subset of those features, can be assigned to a new instance of `Segmenter`, as shown in part 2 of the code. The segmenter needs to be initialized with tuned parameters for p_{off} and p_{on} as discussed in Section 2.2.

Part 3 of the listing demonstrates two methods used to obtain segmentation boundaries and the segmentation state (mode) trajectory (i.e. values of $-$, O, and C from Section 2.2, which can be used to output the segments as CSV or save the resulting segmented waveforms to disk, as shown is parts 4 and 5.

An example standalone application is provided in `examples/segment.cpp`.

B.1.3 Comparison

Finally, individual sound files can be compared against one another as in Section 2.3, handled through two classes, `FeatureComparator` and `SoundComparator`.

`FeatureComparator` instances are used by `SoundComparator` to compare two trajectories of the same feature, and `SoundComparator` finally combines these to obtain likelihood that one set of feature trajectories was constructed by processes represented by the hidden Markov models of another. Below is an example standalone application that demonstrates comparing two or more sounds:

```
#include <iostream>
using namespace std;

#include "sirens//Sirens.h"
#include "sirens/support/matrix_support.h"
#include "sirens/support/string_support.h"
using namespace Sirens;

#include <boost/numeric/ublas/io.hpp>

int main(int argc, char** argv) {
    if (argc < 2) {
        cerr << "Usage: similarity file1 file2 . . . fileN" << endl;
        return 1;
  } else {
        vector<string> files;

        for (int i = 1; i < argc; i++)
            files.push_back(argv[i]);

        // 1. Initialize feature vectors and sounds.
        vector<Sound*> sounds(files.size());
        vector<Loudness*> loudness(files.size());
        vector<TemporalSparsity*> temporal_sparsity(files.size());
        vector<SpectralSparsity*> spectral_sparsity(files.size());
        vector<SpectralCentroid*> spectral_centroid(files.size());
        vector<TransientIndex*> transient_index(files.size());
        vector<Harmonicity*> harmonicity(files.size());
        vector<FeatureSet*> feature_sets(files.size());
        vector<SoundComparator*> comparators(files.size());
```

```
Sound* sound = new Sound();
sound->setFrameLength(0.04);
sound->setHopLength(0.02);

// 2. Iterate through each file, extracting features.
for (int i = 0; i < files.size(); i++) {
    // 2.a. Initialize the sound file.
    sound->open(files[i]);

    int frames = sound->getFrameCount() - 1;
    int spectrum_size = sound->getSpectrumSize();
    int sample_rate = sound->getSampleRate();

    // 2.b. Initialize the features.
    loudness[i] = new Loudness(frames);
    temporal_sparsity[i] = new TemporalSparsity(frames);
    spectral_sparsity[i] = new SpectralSparsity(frames);

    transient_index[i] = new TransientIndex(
        frames,
        spectrum_size,
        sample_rate
    );

    spectral_centroid[i] = new SpectralCentroid(
        frames,
        spectrum_size,
        sample_rate
    );

    harmonicity[i] = new Harmonicity(
```

```
            frames,
            spectrum_size,
            sample_rate
        );

        // 2.c. Initialize the feature set.
        feature_sets[i] = new FeatureSet();
        feature_sets[i]->addSampleFeature(loudness[i]);
        feature_sets[i]->addSpectralFeature(spectral_centroid[i]);
        feature_sets[i]->addSpectralFeature(spectral_sparsity[i]);
        feature_sets[i]->addSampleFeature(temporal_sparsity[i]);
        feature_sets[i]->addSpectralFeature(transient_index[i]);
        feature_sets[i]->addSpectralFeature(harmonicity[i]);

        // 2.d. Extract features.
        sound->setFeatureSet(feature_sets[i]);
        sound->extractFeatures();
        sound->close();

        // 2.e. Initialize SoundComparator for this sound.
        comparators[i] = new SoundComparator(feature_sets[i]);
    }

    // 3. Fill a matrix with log-likelihood values from comparison.
    ublas::matrix<double> likelihood(files.size(), files.size());

    // Compare each sound to itself and the other sound.
    for (int i = 0; i < files.size(); i++) {
        for (int j = 0; j < files.size(); j++) {
            likelihood(i, j) = comparators[i]->compare(
                comparators[j]
```

```
            );
        }
    }

    // 4. Create a symmetric "affinity" matrix.
    ublas::matrix<double> affinity = normalize_affinity(likelihood);

    cout << "Log-likelihood: " << likelihood << endl;
    cout << "Normalized distances: " << affinity << endl;

    // 5. Clean up.
    delete sound;

    for (int i = 0; i < files.size(); i++) {
        delete comparators[i];
        delete feature_sets[i];
        delete loudness[i];
        delete spectral_sparsity[i];
        delete spectral_centroid[i];
        delete transient_index[i];
        delete harmonicity[i];
    }

    return 0;
  }
}
```

Listing B.4: Example comparison of two sounds.

Parts 1 and 2 of Listing B.4 extract features for each of the files specified by the user as command-line arguments. Notice that no additional parameters are necessary for comparison. In Part 3, every possible pair of sounds is compared using the method

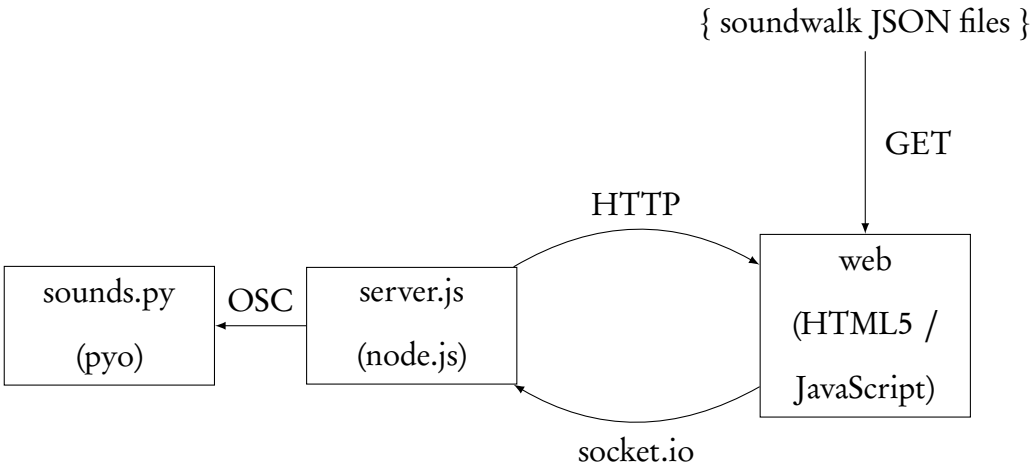


Figure B.1: *Soundwalks* application components.

`SoundComparator::compare`. Note that the emission likelihood calculated and described in Section 2.3 is not symmetrical. As it is sometimes desirable to have a symmetric affinity matrix for various applications, such as the graph retrieval framework outlined in Chapter 3, part 5 creates a symmetric matrix using the semi-metric described in 3.1.1. A full implementation of the above application is available as `examples/similarity.cpp` in *Sirens*.

B.2 *Soundwalks*

The *Soundwalks* contains several components that communicate with each other using a variety of network protocols. In model-view-controller (MVC) terminology, *Soundwalks* can be considered to have two views: a Javascript web application and a Python-based realtime synthesis engine. The two are controlled by a node.js backend server. Soundwalks is distributed as open source software, available at the time of this writing at `http://github.com/plant/soundwalks`. Figure B.1 shows the three major components of Soundwalks. To run Soundwalks from the available source, one simply needs to initialize the server ("node server.js" if node.js has been installed) and the synthesis server

("python sounds.py" once all dependencies have been installed). Then, the application is available as a web application at `localhost:8080`, assuming the default port is used.

Once the web application is loaded, served by `server.js`, it will first asynchronously load a number of JSON files that describe the Soundwalk. These files include GeoJSON formatted GPS paths as well as basic descriptive data about the walks and their associated audio files. An example of GeoJSON is shown below:

```
{
    "type": "FeatureCollection",
    "features": [
        {
            "type": "Feature",
            "properties": {
                "Name": "Path",
                "Description": "Walk to Daley Park"
            },
            "geometry": {
                "type": "LineString",
                "coordinates": [
                    [-111.935155, 33.420156],
                    [-111.934984, 33.419879],
                    ...
                    [-111.929825, 33.407749]
                ]
            }
        }
    ]
}
```

Listing B.5: Portion of an example GeoJSON file.

All communication with `server.js` is then handled using the *socket.io* library using the

WebSocket API. `server.js` forwards information from the web application to the sound synthesis engine in `sounds.py` as the following Open Sound Control (OSC) messages:

1. `/new_walk [filename] [#coordinates]`: When a new walk has been loaded, the synthesis engine is informed of the sound filename to load it, as well as the number of coordinates in its associated GPS path so that it can allocate the necessary memory.

2. `/walk_point [filename] [time offset] [latitude] [longitude]`: When a new walk has been loaded, each GPS point in its GeoJSON attachment is loaded into the synthesis engine with the associated point in time in seconds.

3. `/segments [begin time 1] [end time 1] . . . [begin time N] [end time N]`: When a new walk has been loaded, it can optionally be separated into several independent segments, denoted by their beginning and ending times in seconds. This is useful for grouping multiple recordings into one soundwalk, such as in the case of walks that consist of a number of stationary recordings or walks that have been edited to remove erroneous or sensitive content.

4. `/cursor [latitude] [longitude]`: When the user moves the on-screen cursor, the map latitude and longitude are computed and sent on to the synthesis engine to inform it where to begin synthesizing sounds.

Within *sound.py*, the Python synthesis library *pyo* is used, which allows for realtime synthesis within the Python scripting environment. Python is used for the ease of managing large datasets using the numerical library *numpy*.