Algorithmic Audio Mashups and Synthetic Soundscapes Employing Evolvable Media Repositories

Marinos Koutsomichalis and Björn Gambäck
Department of Computer Science
Norwegian University of Science and Technology
7491 Trondheim, Norway

Abstract

The paper describes a metacreative system for real time algorithmic composition of audio mashups and synthetic soundscapes that pivots on evolvable media repositories, i.e., local pools of related media content that are retrieved, and ever-renovated, over the WWW in an evolutionary fashion. The model also involves a sophisticated soundscape generator, a context aware composer to parametrise it, and a machine listening module performing onset detection and spectral profiling in non-real time. The soundscape generator relies on pattern-based generators to spawn complex and contingent audio sequences in both deterministic and nondeterministic fashions, and is also capable of refined 3D acoustic spatialisation—and multi-channel rendering—with respect to virtual listeners and sonic sources.

1 Introduction and motivation

The aim of the present work is to construe a metacreative system for real time algorithmic composition of audio mashups—i.e., audio collages that comprise pre-recorded music and other found sounds—and synthetic soundscapes, in an algorithmic fashion and with respect to the broader computer/experimental music tradition. The system is based on dedicated hardware and a series of embedded software components implementing an evolvable media repository (EMR), and miscellaneous modal and cross-modal synthesis modules for the generation of images and video, in addition to the audio. The implementation is designed so that the system autonomously generates multichannel audio to be used in the context of sound and media art installations.

The system is meant both as digital art of an experimental kind, and as a hands-on endeavour to explore and probe the materiality of user-generated content (UGC) repositories of audio, and of the technologies we rely upon to interact with them. Accordingly, the model draws on the broad tradition of material-driven and epistemologically concerned art (Koutsomichalis 2015; Gough 2005; Barrett and Bolt 2013) and primarily supports the exploration, rather than the integration, of existent internet-based UGC databases. In contrast to other approaches to automatic audio composition—e.g. (Collins 2012)—and music mashups—e.g. (Davies et al. 2013; Tokui 2008)—the system does not select content that may best fit some particular aesthetic or functional goal, but rather aims at foregrounding and exposing all sorts of disparate aesthetic, and non-aesthetic, orderings that the available data and their associated meta-data may themselves imply.

The rest of the paper zooms in the technical specifics of the proposed model, and on its implementation. The next section outlines related research. Then Section 3 explains the model in detail, elaborates on its constituent parts, and gives some implementation details, while Section 4 discusses the resulting audio, and the model's performance. Directions for future research and concluding remarks follow.

2 Related work

There are several documented approaches to algorithmic audio mashup generation ranging from intelligent semi-automatic composers (Davies et al. 2013; Griffin, Kim, and Turnbull 2010), to sophisticated web-based collective systems (Tokui 2008). However, the vast majority of existing systems concern popular music (Gunkel 2011), and are either meant to be utilised in some functional context, or result in 'toy' compositions. In particular, literature on audio mashup generation often overlook a long train of real, and non-real, time composition practices concerning the creative, and metacreative, use of existent and *ad hoc* repositories of audio/music (Koutsomichalis 2016).

The present work concerns the production of abstract synthetic audio in a purely metacreative fashion, focusing on the particular orderings the data themselves bring forth, as well as on the semantic cross-associations that are forged among them. As such it is indifferent to beat-synchronicity, pitch/time shifting techniques, user interaction, and similar traits that typically concern functional systems for music mashups, but is instead relevant to, and draws upon: (a) soundscape studies (Wrightson 2000), and an array of relevant composition practices that are broadly concerned with the juxtaposition of 'found' sounds of natural, wildlife, urban, or other origin (Truax 2002; Westerkamp 2002; Drever 2002; Koutsomichalis 2013); (b) those particular traits in the broad 'acousmatic', electroacoustic, and computer music traditions that pivot on the manipulation of found sounds to generate (abstract) narratives that are not necessarily rhythmical or beat-synchronous. It should be, nevertheless, noted that 'synthetic soundscapes' does not nevessarily refer to recorded sounds of natural or other origin; we rather refer to abstract compositions that, like most real-life soundscapes, comprise sonic events that are widely distributed, and often move, in acoustic space.

Automatic/algorithmic soundscape composition has previously been addressed within augmented reality (Warusfel and Eckel 2004), the video-game industry,[1] and other functional contexts (Misra, Wang, and Cook 2007; Valle, Lombardo, and Schirosa 2010). SoDA, Sound Design Accelerator (Valle et al. 2014) concerned the algorithmic composition of soundscapes with respect to a series of user-defined natural language queries describing the content and features in terms of desired constituent sound events/sources. SoDA was originally intended as a tool for sound designers to use in real-life (pre-)production contexts, but one of its constituent modules, Soundscape Generator (SSG) (Koutsomichalis and Valle 2014), was designed to facilitate context-agnostic multichannel soundscape composition in both real and non-real time and is therefore potentially applicable to arbitrary functional and artistic endeavours.

The present work relies on techniques first introduced in SSG, but here the focus is on exploring the repositories of interest in their entirety and simultaneously. This brings forth the question of how to navigate databases that are astronomically big in some structured fashion, possibly revealing implicit endogenous narratives on the way. The question of querying, or otherwise navigating, the available data space is central to any project concerning large pools of media content—it is typically with respect to how data are retrieved that they acquire value in some artistic or functional context. For instance, in the case of Napier's *Pam*[2] series of works, all queries pivot on Pam Anderson, who in this way becomes the thematic axis of the project. Manovich's *Selfiecity* and *On Broadway* (Manovich 2015) rely on more sophisticated queries, exploring Instagram for photos that ascribe a particular structural—i.e., being a 'selfie'—or geographical—i.e., being shot in Broadway Street—attribute. In the case of SoDA, or (Koutsomichalis and Gambäck 2018) concerning computational solid modelling employing media retrieved from UGC repositories, natural language queries (NLQ) are performed automatically with respect to user-defined text input.

Of particular relevance here is *Audio Metaphor* (Thorogood and Pasquier 2013), a metacreative system producing sound-art from short natural language tokens provided by a user or received from Twitter. The language queries retrieve labelled audio recordings from FreeSound[3] that are, subsequently, segmented by a supervised machine learner trained on data from human perceptual classification experiments and, finally, processed and juxtaposed employing a modelled composition schema. *Audio Metaphor* has been successfully

[1]`http://anasynth.ircam.fr/home/english/projects/anr-project-physis`

[2]`http://www.marknapier.com/portfolio/pam/`

[3]`www.freesound.org`

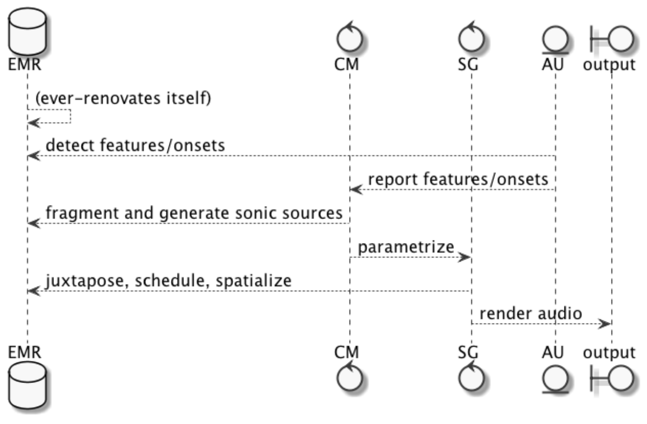


Figure 1: Interaction schemata between constituent modules (the nodes on the top are identical to the ones at the bottom)

used in conjunction with *Re:Cycle* (Bizzocchi 2011)—a system for real-time generative video synthesis—generating audio that may relate to the video track in a number of different fashions (Eigenfeldt et al. 2014).

The present work shares some common ground with *Audio Metaphor*, both regarding method and scope, but relies on a much more sophisticated database management system, creating an evolvable media repository (EMR) in order to crawl several UGC repositories of interest (FreeSound, YouTube, SoundCloud, Wikipedia, etc.) for media content. The repository ever-populates itself contingently, while respecting, to a varying degree, the semantic associations suggested by both content and meta-data. It maps language queries to graph-based genotypic representations, and relies on modularly attached media 'comprehenders' to make sense of the retrieved content and associated meta-data. In contrast to *Audio Metaphor*, the technical realisation pivots on ambisonics (Malham and Myatt 1995) for 3D audio spatialisation and may thus deliver truly immersive multichannel audio in real-time.

3 Method and implementation

As shown in Figure 1, the system comprises four modules:

EMR, evolvable media repository: crawling—in an evolutionary fashion—YouTube, Soundcloud, Freesound, and other online repositories for related media content.

AU, audio understanding: performing onset-analysis and spectral feature extraction.

SG, graph-based Soundscape Generator: real-time temporal scheduling, localisation in 3D acoustic space, and appropriate multichannel encoding/decoding.

CM, composition module: generating contingent sonic sources in various flavours from the retrieved media content, and parametrising SG accordingly.

The figure illustrates the ways in which modules interact with one another. The following subsections scrutinize the modules and their interactions in detail. With the exception of EMR, which is implemented in Python, all modules have been coded in SuperCollider (McCartney 2002).

3.1 Evolvable media repository, EMR

Evaluation and selection are almost unanimously accepted as essential steps in every Evolutionary algorithm (EA). Despite being debatable to what extent such solutions succeed in generating genuine artistic value in real-life contexts (McCormack 2005; Bown and McCormack 2010), the vast majority of art-oriented EAs tend to rely on fitness functions of some sort—as shown, for instance, in (Johnson 2012). Our EMR, nevertheless, does not employ fitness function and does not rely on selection processes. It should be rather thought of as an *ad hoc* solution pivoting on mutation for the sake of variance, and in order to generate new and original content for the sake of it.

EMR encodes genotypic space through undirected graphs of natural language tokens on the form (V, E), with V being a set of vertices $\{v_1, v_2, \ldots, v_n\}$, $n \in \mathbb{Z}^+$, $v_n \in U^*$ and $v_n \neq \emptyset$—U^* comprising all possible word sequences over the unicode character set; and E a (possibly empty) set of pairs $\{v_p, v_k\}$ for some $v_p, v_k \in V$ and $v_p \neq v_k$. The EMR algorithm can be formulated as:

$$\mathrm{G_n} = \begin{cases} \bigcup \Phi_\Lambda(\bigcup_{i=1}^{n} \lambda_i(\mathrm{G_{n-1}})) & : n \in \mathbb{N}_{>1} \\ \langle S \rangle & : n = 1 \end{cases} \tag{1}$$

where $\langle S \rangle$ is the 'seed', i.e., the genetic material of the initial population; $\Phi_\Lambda : P_\Lambda^+ \to G^+$ a 'comprehender' relation mapping a phenotypic element—in this particular context, phenotypes should be thought of as the resulting pools of media files, $P \in W$ (with W representing the WWW) of type Λ back to genotypic space; and $\lambda_i : G^+ \to P^*$ a 'crawler' relation retrieving content from some online repository and keeping it locally. The resulting phenotype comprising the retrieved media files after k iterations of the algorithm is:

$$P_k = \bigcup \lambda_i(\mathrm{G_k}) \tag{2}$$

Note that the vertices in a genome need not necessarily be meaningful natural language tokens, the various natural languages being just a subset of U^*. The genome may properly encode and hierarchically represent arbitrary semantic, lexical and symbolic relationships that may arise naturally in different contexts and irrespective of particular human or machine languages, codes, and esoteric jargons.

Implementing an EMR is then a question of defining a representation for G_n, designing appropriate crawlers λ_i for the repositories of interest, and appropriate comprehenders Φ_Λ needed to understand them. In our implementation, all crawlers inherit from an `AbstractCrawler` class and are required to implement an iterator over the results, as well as methods to query for, retrieve, and post-process content, and retrieve metadata and associated para-texts. Implementing such methods is of varying complexity, depending on the particular APIs and the authorization specifics at play. We have implemented crawlers for YouTube, SoundCloud, Freesound, Wikipedia, MLDB, WordNet, Flickr and ConceptNet, that retrieve audio content (extracted from video in the case of YouTube), associated meta-data such as text descriptions, tags and user comments, encyclopedic text entries, music lyrics, synonyms, and related terms. Retrieving also non-audio content (such as text or image) makes sense here since that kind of content may still be 'comprehended', in this way contributing semantically to the evolution cycle. In our implementation, all comprehenders inherit from an `AbstractComprehender` class and are required to implement methods to prepare and understand data, and to take care of any cleaning tasks. We have implemented three types of comprehenders: an image comprehender employing the Inception-v3 (Szegedy et al. 2016) network and trained on the ImageNet (Deng et al. 2009) LSVRC-2012 challenge data set; a text comprehender that relies on a RAKE (Rose et al. 2010) algorithm for natural language understanding to have the input reduced into a series of associated queries; and a trivial 'tag' comprehender that merely assembles a graph out of every available tag token.

All comprehenders result in fully connected weighted graphs that may be merged together into singleton representations. In such merged graphs, queries are interconnected with (a) all other queries generated by the same comprehender for the same media file, and (b) all other queries that any other comprehender has deemed relevant to this particular query in the context of some other file. Iteration over some genotype G respects such associations, so that every vertex output u_n is either the next highest ranking vertex connected to u_{n-1}, or the next highest ranking vertex that has not already appeared in the output. Then, as crawlers iterate through a genome, they draw queries in such a fashion.

EMR is modular and may be parameterised. It is possible to select what crawlers and comprehenders should be utilised, how long the retrieved audio files should be kept, what is the maximum number of files that should be retrieved per crawler or per iteration cycle, and a number of other parameters.

3.2 Audio Understanding, AU

The audio understanding module utilises a series of SuperCollider-specific machine listening unit generators (Collins 2011) to perform onset analysis and feature extraction in non-real time. During onset analysis, a vector is generated comprising the times throughout the duration of the piece when some kind of change (e.g., in pitch, rhythm, or timbre) has been detected. Onsets are found by a SuperCollider implementation of the 'adaptive whitening' onset detection algorithm (Stowell and Plumbley 2007), and utilised by the composition module when fragmenting the files in constituent chunks of audio. During feature extraction, AU analyses the audio spectrum in regular time intervals T—a user-defined argument—capturing analysis snapshots throughout the duration of the input file. The particular features of interest herein are the centroid (the weighted mean frequency of the spectrum), the spread (the magnitude-weighted variance of the spectrum), and the complexity (a value representing how 'complex' the spectrum is, with the extremes being white noise and a single tone). AU results in a matrix of features:

$$\mathbf{\Phi} = \begin{bmatrix} \mathbf{c} \\ \mathbf{u} \\ \mathbf{s} \end{bmatrix} = \begin{bmatrix} c_0 & c_1 & \dots & c_n \\ u_0 & u_1 & \dots & u_n \\ s_0 & s_1 & \dots & s_n \end{bmatrix} \tag{3}$$

where $\mathbf{c}, \mathbf{u}, \mathbf{s}$ represent the resulting vectors for centroid, complexity, and spread, respectively, and with $n = \lfloor \frac{L}{T} \rfloor$, L being the length of the audio file. In the implementation, all features are normalised to the $[0, 1]$ range.

3.3 Soundscape generator, SG

The Soundscape generator is a heavily modified fork of SSG originally used in SoDA, and supports:

- 3D sound localisation that takes into account the positioning of a virtual listener and of the various virtual sonic sources, as well as their sizes (spatial footprints), but not a sound-zone's acoustic features as the original SSG.
- Deterministic and nondeterministic temporal scheduling, and localisation of individual sounds in acoustic space.
- Deterministic and nondeterministic generation of complex sound events out of atomic sound samples.
- Modelling of sound events, the virtual positioning of which may move deterministically in 3D space.
- Multipurpose audio decoding to arbitrary (possibly multichannel) speaker configurations.
- Real-time (only, unlike SSG) rendering of sonic sources on-the-fly as they are added to or removed from the internal synthesis graph.

The SG renderer expects as arguments the 3D bounds of the acoustic space, a 'listener' object (which may be fixed or movable in acoustic space), and a 'decoder' object that manages the desired output format. SG relies on an ambisonic spatialisation algorithm (Malham and Myatt 1995), so that, given the appropriate decoder, the same audio stream may be decoded to any of the standard formats such as mono, stereo, or quad, but also to custom, arbitrary periphonic, or holophonic, speaker configurations. Once the renderer has been initiated, 'sonic sources' may be added to, or removed from, the internal synthesis graph. Three types of such sources are currently employed:

Atmosphere: non-directional sonic ambience.

Fixed Sound: fixed directional sources.

Ambulatory Sound: an ambulatory directional source.

All sonic sources are associated with pattern-based audio 'sequences' of arbitrary complexity, and with some repetition pattern. Directional sonic sources have two additional arguments: the size of the virtual sonic object they represent and its spatial coordinates in acoustic space. These are either a fixed point (fixed sound), or three envelopes representing trajectories for each spatial dimension (ambulatory sound).

Audio sequences in SG are specified employing the SuperCollider's inbuilt list-pattern generators (Kuivila 2011) that include representations for linear sequences, random selections from lists, probability-based number generators, random walks, and other similar mathematical constructs that provide a conceptually straightforward and highly expressive way to model streams of values. Moreover, such patterns may be chained or nested recursively, allowing for a compact notation of very complex behaviour.

An audio sequence S is formally defined as a tuple of relations ($\sigma \colon \mathbb{N}_{<k} \to H, p \colon \mathbb{N}_{<i} \to \mathbb{Q}^+$) representing list patterns of arbitrary complexity, with σ returning memory locations holding audio content—H comprising all available memory slots—and p returning time durations. Note that the two relations are only defined within the intervals $[0, k)$ and $[0, i)$, respectively, for some k and i that may or may not be equal, and that may be infinite. Playing back some audio sequence is to evaluate σ and p with integer increments $0, 1, 2, \dots, n$, with $n \leq \min(k, n)$, at irregular time intervals and following the output of p, so that audio stored in $\sigma(n)$ is performed for $p(n)$ number of seconds and, when done, audio stored in $\sigma(n+1)$ is performed for $p(n+1)$ seconds, etc. As currently implemented, whenever $p(n) = 0$, $\sigma(n)$ is played back in its entirety.

Once a source is added to the synthesis graph, it is immediately scheduled for play-back with respect to an associated pattern-based relation $q : \mathbb{N}_{<k} \to \mathbb{Q}^+$, returning values representing silent pauses before the first, and between all subsequent, appearances of their encapsulated audio sequence. Performing the latter is to route its audio output—which as is generated contingently in real time—first to an appropriate localisation synthesizer and then to the output decoder. Localisation takes into account the differences $\Delta_x(t), \Delta_y(t), \Delta_z(t)$ between the listener's positioning and that of the sonic source for each dimension in some discrete time $t \in \mathbb{Z}$ (measured in samples), as well as the source's associated size r.

The encoding is better understood in terms of a radius $\rho(t) = \sqrt{\Delta_x(t)^2 + \Delta_y(t)^2 + \Delta_z(t)^2} - \frac{r}{2}$, together with an azimuth angle $\phi(t) = \arctan \frac{\Delta_y(t)}{\Delta_x(t)}$ and a zenith angle $\theta(t) = \arccos \frac{\Delta_z(t)}{\rho(t)}$. Then the b-format ambisonic signal is:

$$\begin{bmatrix} w(t) \\ x(t) \\ y(t) \\ z(t) \end{bmatrix} = aA(t) \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \cos\theta\cos\phi \\ \sin\theta\cos\phi \\ \sin\phi \end{bmatrix} \tag{4}$$

where a is a user-defined amplitude factor. In the case of stereo signals, the right part of Equation 4 becomes:

$$aA_l(t) \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \cos\theta\cos\phi \\ \sin\theta\cos(\phi + \frac{pi}{2}) \\ \sin\phi \end{bmatrix} + aA_r(t) \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \cos\theta\cos\phi \\ \sin\theta\cos(\phi - \frac{pi}{2}) \\ \sin\phi \end{bmatrix}$$

where A_l, A_r are the left and right channels, respectively.

Then, the renderer mixes together the resulting soundfields $[w_i(t), x_i(t), y_i(t), z_i(t)]$ for all i active sonic sources, and routes their sum to the user-defined decoder D which handles the actual multichannel audio output vector $\mathbf{o}$.

$$\mathbf{o} = D\Big(\sum_{1}^{i} \big[w_i(t), x_i(t), y_i(t), z_i(t)\big]\Big) \tag{5}$$

Decoders of various kinds may be employed, mapping the resulting ambisonic sound field to an appropriate array of audio signals to drive one, or more, speakers in mono, stereo, 5.1, 7.1, quad, octaphonic, or some other arbitrary periphonic or pantaphonic configuration.

3.4 Composition module, CM

The composition module takes into account the origin of the retrieved audio files, their spectral profiles as generated by AU, and a series of predefined composition rules. Figure 2 illustrates the activity cycle. For whatever new files found in the EMR, CM asks AU to perform an onset analysis, and then fragments the original file into short, longer or long chunks with respect to both their origin and duration. It then appends them to audio sequences, calculates the spatialisation coefficients, spawns sonic sources, and adds them to the synthesis graph for immediate reproduction. Following the requirement for a system that can be publicly exhibited in some real-life setting, the waiting patterns for all sonic sources are such that they repeat themselves after a few minutes—in this way guaranteeing that audio playback is continuous. An auxiliary mechanism featuring a priority queue is constantly monitoring the loop, so that old sources are removed from the synthesis graph, and so that used audio files and chunks are deleted from the EMR, once a sufficient number of new files have been downloaded and processed.

For content retrieved from SoundCloud (i.e., music), or lasting less than 20 seconds, it makes little sense to employ ambulatory spatialisation, to reproduce it in its original form, or to spawn more than one sequence per audio file — at least not in system's current aesthetic setting. Accordingly, 70% of the times a random walk is performed (i.e, after some chunk has finished playing back, the algorithm may continue with the antecedent one, the precedent one, or the same again) and 30% of the times the chunks are scrambled stochastically. Note that such sequences are defined by patterns and performed dynamically, so their repetitions will not result in the exact same audio.

The CM cannot make any assumption on the kind of content retrieved from FreeSound or YouTube, which may practically concern everything. As shown in figure 2, such content is fragmented in longer chunks, with respect to the detected onsets and their original duration, and individual sonic sources are only generated for some of them. Those sources are scheduled for reproduction in some random time in the immediate future (i.e., a couple of minutes at most). Thus, several chunks originating from the very same file may happen to sound simultaneously, resulting both in a thicker soundscape and in a certain kind of spatiotemporal deconstruction, which is more appropriate when the content of the file is not known. As also shown in the figure, the CM takes into account the duration of the original file as well as the spectral profile of each fragment in order to decide whether the resulting source is static, ambulatory, or non-directional.

Sonic sources are then spatialised so that they are well spread in space in a coherent and compositionally meaningful way. Localisation of a particular source is carried out by a linear mapping from its associated spectral profile in $[0, 1]^3$ to a Cartesian space C^3 that spans from the origin $[0, 0, 0]$ to a user-defined $[X_m, Y_m, Z_m]$. First, the vectors $\boldsymbol{\rho} = \frac{\frac{X_m+Y_m}{4}\frac{\mathbf{s}+\mathbf{c}}{2}}{\frac{X_m+Y_m}{4}}, \boldsymbol{\phi} = \frac{2\pi\frac{\mathbf{c}+\mathbf{u}}{2}}{2\pi}, \boldsymbol{\zeta} = \frac{Z_m\frac{u+s}{2}}{Z_m}$, are calculated. These are then mapped to a matrix comprising vectors in C^3, as follows:

$$\begin{bmatrix}\mathbf{x}\\ \mathbf{y}\\ \mathbf{z}\end{bmatrix} = \begin{bmatrix}\boldsymbol{\rho}\cos\boldsymbol{\phi}\\ \boldsymbol{\rho}\sin\boldsymbol{\phi}\\ \boldsymbol{\zeta}\end{bmatrix} \tag{6}$$

In the case of some ambulatory source, and employing linearly interpolation, a continuous trajectory in 3D space can be represented as a continuous function of time $t \in \mathbb{R}^+$.

$$f(t) = \begin{bmatrix}x(t)\\ y(t)\\ z(t)\end{bmatrix} = \begin{bmatrix}\rho_r\cos\phi_r + (t-r)\frac{x_k-x_r}{k-r}\\ \rho_r\cos\sin_r + (t-r)\frac{y_k-y_r}{k-r}\\ \zeta_r + (t-r)\frac{z_k-z_r}{k-r}\end{bmatrix} \tag{7}$$

where $r, k \in \mathbb{N}$, are column indices in $\boldsymbol{\Phi}$, representing discrete measurements in T second intervals, and $r \le t \le k$. The actual model (of course) employs a discrete time, sampled, version of $f(t)$.

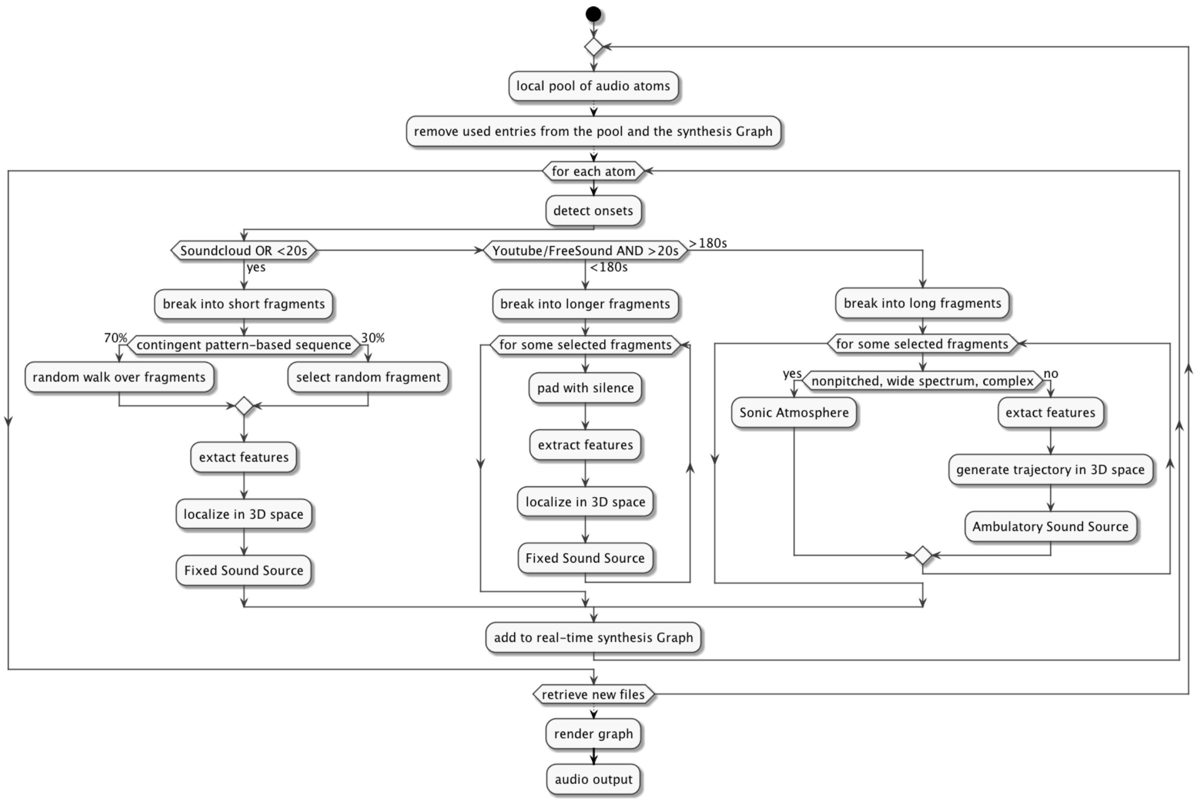
IN the case of fixed sources, the quadratic means of $\mathbf{c}, \mathbf{u}, \mathbf{s}$ are taken, so that $\mathbf{c} \mapsto \sqrt{\frac{1}{\dim\mathbf{c}}\sum_{i=0}^{\dim\mathbf{c}} c_i^2}$, $\mathbf{u} \mapsto \sqrt{\frac{1}{\dim\mathbf{u}}\sum_{i=0}^{\dim\mathbf{u}} u_i^2}$, and $\mathbf{s} \mapsto \sqrt{\frac{1}{\dim\mathbf{s}}\sum_{i=0}^{\dim\mathbf{s}} s_i^2}$, and so that all $c, u, s, \rho, \phi, \zeta, x, y, z$ collapse to scalars. The fixed position is then given by Equation 6, replacing all the vectors with their equivalent scalars.

4 Discussion

Example output compositions rendered in stereo may be retrieved from `https://tinyurl.com/emrcomp`. The audio files concern excerpts from longer real-time recordings lasting several minutes, and are named according to the seed that initiated the EMR, and a note stating whether they concern the start, or some subsequent part, of the recording.

As already explained, our original motivation concerned a system to be used in an installation context and intended to demonstrate how found content may be creatively explored. Sound installations, however, pose certain aesthetic and technical requirements that the model has to address. Accordingly, the given results reflect a series of design decisions, configurations and optimisations that have been applied to the model.

First and foremost, the model has been optimised to generate audio in a continuous and seamless manner. In the context of some installation artwork, audiences are expected to enter and leave at their own discretion, typically spending just a few minutes of active listening. Hence it is more important to guarantee that output of some complexity is present at any given moment, than to focus on the time-based development of texture and content. The system has therefore been designed so that it will not remain silent for more



local pool of audio atoms
remove used entries from the pool and the synthesis Graph
for each atom
detect onsets
Soundcloud OR <20s
yes
Youtube/FreeSound AND >20s
>180s
<180s
break into short fragments
break into longer fragments
break into long fragments
70%
contingent pattern-based sequence
30%
random walk over fragments
select random fragment
extact features
localize in 3D space
Fixed Sound Source
for some selected fragments
pad with silence
extract features
localize in 3D space
Fixed Sound Source
for some selected fragments
yes
nonpitched, wide spectrum, complex
no
Sonic Atmosphere
extact features
generate trajectory in 3D space
Ambulatory Sound Source
add to real-time synthesis Graph
retrieve new files
render graph
audio output

Figure 2: Activity cycle

than a few seconds and so that no definite start or end is implied. To make it aesthetically meaningful to keep listening for more than a few minutes, audio material is recycled in a nondeterministic fashion, employing pattern-based tactics, so that whenever materials already used are repeated, they are ordered differently temporarily and juxtaposed with respect to one another.

It should be added that, in some real-life installation setting, it is practically impossible to guarantee that new audio is retrieved in short, or regular, intervals. The frequency in which EMR draws material from WWW depends on a series of factors such as network speed, CPU power (e.g., in the case of YouTube, CPU-intensive processing is necessary to pre-process the audio file), the size of files to download, hard disk speed, and other factors. Recycling of audio content is, therefore, both inevitable and desired.

The choices regarding the particular repositories the model pivots on, and the particular ways in which it deconstructs and reconstructs the retrieved audio follow the artistic reasoning of the broader project. Here the focus has been on three of the most famous and widely used repositories for peers to share music, and on arbitrary sound recordings so that the audio/music content of the WWW is explored in its true volume and variety. De-/re-construction is then performed by means of random walks, scrambling, fragmentation, and concatenation, so that the system explores the raw material in a variety of different ways, often resulting in different kinds of textures, and so that the original structure and content of the retrieved files are maintained to a varying degree.

Several other parameters control the maximum number of files to be simultaneously present in the synthesis graph, the approximate number of audio fragments to consider, their maximum and minimum duration, the size and geometry of the virtual acoustic space, and, most importantly, the percentage by which the repositories of interest contribute to the EMR. All sorts of different settings of these parameters have been experimented with, but preference has been given to YouTube and SoundCloud, since the applications currently studied in particular concern popular music and culture. In the given examples the maximum numbers of files/fragments to consider range from 4 files and 4 fragments per file, to 8 files and 10 fragments per file. Larger values could certainly make sense in other aesthetic contexts, but currently the main concern has been to make explicit (or at least imply), the origin of, and possibly any special connotations of, any sound, music, and spoken word that appear in the output—this would be hard, if not impossible, to achieve in an overly dense soundscape. Note also that, despite the Sound Generator being capable of complex and sophisticated audio localisation patterns, rather restrained settings have currently been used in order to avoid distraction.

From a compositional point of view, the output of the model ranges from lengthy audio mashups, to abstract sound synthesis, to clear-cut music gestures, and even to narration—often in the course of the very same session. account for two very different textures.

Rhythmic sections, semi-diegetic sections, and referential gestures, occasionally emerge. Albeit being secondary affairs from a technical viewpoint (priority been given to having continuous audio in the output even in the case of some network failure), they are desired as far as the particular artistic rationale of the project is concerned. It should be emphasised that such phenomena are not at all haphazard, but depend on both the particular ways in which the CM manipulates the retrieved data (i.e., allowing fragmentary and contingent reproduction, but not to the extent that they are unrecognizable) and, most importantly, to how EMR is configured. Here, for instance, EMR takes into account music lyrics, encyclopedic entries, and semantic associations, in addition to any meta-data associated with the retrieved audio files. In this fashion, the system may better resolve, and represent, implicit thematic relationships, resulting in audio files that relate to one another in varying ways.

5 Conclusion and future work

The paper has described a system for the real time algorithmic composition of audio mashups and synthetic soundscapes, employing an evolvable media repository, a composition module, and a sophisticated soundscape generator. If properly configured, the system produces audio that does not collapse to arbitrary mashups or to the same ever-repeating textures, but instead is characterised by ever-contingent temporal development both insofar as content and texture is concerned. The output of the system may range from monolithic audio collages, to abstract synthesis, to music narratives, to clear-cut music gestures, and almost diegetic hybrids, by virtue of manipulating ever-renovated related audio content.

The system is meant to be used in the context of a broader media installation artwork and, accordingly, it is a strong requirement that audio may be generated continuously and seamlessly, even in the case of network failure. Hence, the configuration is such that retrieved audio is ever-recycled until the EMR delivers new content. However, the design also guarantees that whenever the very same audio content is used, it will always result in contingent and more or less different temporal orderings, thus revealing alternative creative perspectives on how the given audio may be re-used. In addition, it is possible to specialise the EMR so that it zooms in on repositories having particular kinds of content, and so that additional resources (e.g., encyclopedic entries or semantic associations) are employed in order to resolve and represent thematic relationships, resulting in audio content with a higher likelihood to spawn music narratives, gestures and even diegesis. The system is potentially of use in a broad range of computer music, sound, and media art contexts that concern the creative re-appropriation of found content—at the very least as a source of inspiration, but also as a concrete algorithmic paradigm for others to re-configure or extend. So far, its creative potential has only been explored in the context of some particular artistic project and with respect to the requirements it imposes. Still, the model can be made to fit several other composition paradigms and genres of various shorts—as is, or with minor modifications. For instance, it is trivial to extend the system so that it generates beat-synchronous mashups. It could be also used as is in an off-line and interactive fashion, automatically generating contingent collages and microscopic narratives that could later be used selectively in the context of some studio-based electroacoustic music composition.

The model already satisfies the requirements of the project, given its experimental scope and speculative aesthetics. We, nevertheless, intend to extrapolate the paradigm of EMR-driven synthesis to other media, designing and implementing algorithmic composers producing video, image, text, and other artefacts, in a generative fashion. We are currently working towards designing synthesizers generating 3D-printable solid models, and images.

Acknowledgements

This work was carried out when the first author was supported by an ERCIM "Alain Bensoussan" Fellowship.

References

Barrett, E., and Bolt, B. 2013. *Carnal knowledge: towards a new materialism through the arts*. New York, NY: Ib tauris.

Bizzocchi, J. 2011. Re: Cycle-a generative ambient video engine. In *Proceedings of the International Conference on Entertainment Computing (Vancouver, Canada)*, 354–357. Berlin, Heidelberg: Springer.

Bown, O., and McCormack, J. 2010. Taming nature: tapping the creative potential of ecosystem models in the arts. *Digital Creativity* 21(4):215–231.

Collins, N. 2011. Machine listening in supercollider. In Wilson, S.; Cottle, D.; and Collins, N., eds., *The SuperCollider Book*. Cambridge, MA: The MIT Press. 439–462.

Collins, N. 2012. Automatic composition of electroacoustic art music utilizing machine listening. *Computer Music Journal* 36(3):8–23.

Davies, M. E.; Hamel, P.; Yoshii, K.; and Goto, M. 2013. Automashupper: An automatic multi-song mashup system. In *Proceedings of the 14th International Society for Music Information Retrieval Conference (Curitiba,Brazil)*, 575–580. ISMIR.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 248–255. IEEE.

Drever, J. L. 2002. Soundscape composition: the convergence of ethnography and acousmatic music. *Organised Sound* 7(1):21–27.

Eigenfeldt, A.; Thorogood, M.; Bizzocchi, J.; and Pasquier, P. 2014. Mediascape: Towards a video, music, and sound metacreation. *Journal of Science and Technology of the Arts* 6(1):61–71.

Gough, M. 2005. *The artist as producer: Russian constructivism in revolution*. Oakland, CA: Univ of California Press.

Griffin, G.; Kim, Y. E.; and Turnbull, D. 2010. Beat-sync-mash-coder: A web application for real-time creation of beat-synchronous music mashups. In *Proceedings of 35th IEEE International Conference on Acoustics Speech and Signal Processing (Dallas,TX)*, 437–440. Piscataway, NJ: IEEE.

Gunkel, D. J. 2011. Audible transgressions: Art and aesthetics after the mashup. In Gournelos, T., and Gunkel, D. J., eds., *Transgression 2.0: Media, Culture, and the Politics of a Digital Age*. New York, NY: Continuum. 42–56.

Johnson, C. 2012. Fitness in evolutionary art and music: what has been used and what could be used? *Evolutionary and Biologically Inspired Music, Sound, Art and Design* 129–140.

Koutsomichalis, M., and Gambäck, B. 2018. Generative solid modelling employing natural language understanding and 3d data. In A., L.; J., R. C.; and Ekárt, A., eds., *Computational Intelligence in Music, Sound, Art and Design. EvoMUSART 2018.*, volume 10783 of *Lecture Notes in Computer Science*. New York, NY: Springer.

Koutsomichalis, M., and Valle, A. 2014. Soundscapegenerator: soundscape modelling and simulation. In *Proceedings of the 20th Colloquium on Music Informatics (Rome, Italy)*, 65–70. Università IUAV di Venezia.

Koutsomichalis, M. 2013. On soundscapes, phonography and environmental sound art. *Journal of sonic studies* 4(1).

Koutsomichalis, M. G. 2015. *A hypermedia and Project-based approach to Music, Sound and Media Art.* Ph.D. Dissertation, De Montfort University, Leicester, U.K.

Koutsomichalis, M. 2016. Catalogue aesthetics: Database in and as music. In Kostagiolas, P.; Martzoukou, K.; and Lavranos, C., eds., *Trends in Music Information Seeking, Behavior, and Retrieval for Creativity*. Hershey, PA: IGI Global. 258–277.

Kuivila, R. 2011. Events and patterns. In Wilson, S.; Cottle, D.; and Collins, N., eds., *The SuperCollider Book*. Cambridge, MA: The MIT Press. 179–206.

Malham, D. G., and Myatt, A. 1995. 3-d sound spatialization using ambisonic techniques. *Computer music journal* 19(4):58–70.

Manovich, L. 2015. Exploring urban social media: Selfiecity and on broadway. Retrieved May 12 2018 from `http://manovich.net/content/04-projects/083-urbansocialmedia/manovich_exploring_urban_social_media_edit.pdf`.

McCartney, J. 2002. Rethinking the computer music language: Supercollider. *Computer Music Journal* 26(4):61–68.

McCormack, J. 2005. Open problems in evolutionary music and art. *Applications of Evolutionary Computing* 428–436.

Misra, A.; Wang, G.; and Cook, P. 2007. Musical tapestry: Re-composing natural sounds. *Journal of New Music Research* 36(4):241–250.

Rose, S.; Engel, D.; Cramer, N.; and Cowley, W. 2010. Automatic keyword extraction from individual documents. In Berry, M. W., and Kogan, J., eds., *Text Mining: Applications and Theory*. Chicester, U.K.: John Wiley & Sons. 1–20.

Stowell, D., and Plumbley, M. 2007. Adaptive whitening for improved real-time audio onset detection. In *Proceedings of the International Computer Music Conference (Copenhagen, Denmark)*, 312–319. Copenhagen, Denmark: Re:New - Digital Arts Forum.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2818–2826.

Thorogood, M., and Pasquier, P. 2013. Computationally created soundscapes with audio metaphor. In *Proceedings of the Fourth International Conference on Computational Creativity (Sydney, Australia)*, 1–7.

Tokui, N. 2008. Massh!: a web-based collective music mashup system. In *Proceedings of the third ACM International Conference on Digital Interactive Media in Entertainment and Arts (Athens, Greece)*, 526–527. New York, NY: ACM.

Truax, B. 2002. Genres and techniques of soundscape composition as developed at simon fraser university. *Organised sound* 7(1):5–14.

Valle, A.; Armao, P.; Casu, M.; and Koutsomichalis, M. 2014. Soda: A sound design accelerator for the automatic generation of soundscapes from an ontologically annotated sound library. In *Proceedings of the International Computer Music Conference — Sound Computer Conference (Athens, Greece)*, 1610–1617. San Francisco, CA: International Computer Music Association.

Valle, A.; Lombardo, V.; and Schirosa, M. 2010. Simulating the soundscape through an analysis/resynthesis methodology. In Ystad, S.; Aramaki, M.; Kronland-Martinet, R.; and Jensen, K., eds., *Auditory Display: 6th International Symposium, CMMR/ICAD 2009, Copenhagen, Denmark, May 18-22, 2009, Revised Papers*. Berlin, Heidelberg: Springer. 330–357.

Warusfel, O., and Eckel, G. 2004. Listen-augmenting everyday environments through interactive soundscapes. Retrieved February 24 2018 from `http://resumbrae.com/vr04/warusfel.pdf`.

Westerkamp, H. 2002. Linking soundscape composition and acoustic ecology. *Organised Sound* 7(1):51–56.

Wrightson, K. 2000. An introduction to acoustic ecology. *Soundscape* 1(1):10–13.