DESCRIPTOR-BASED SOUND TEXTURE SAMPLING

Diemo Schwarz
Ircam–CNRS STMS
Paris, France

Norbert Schnell
Ircam–CNRS STMS
Paris, France

ABSTRACT

Existing methods for sound texture synthesis are often concerned with the extension of a given recording, while keeping its overall properties and avoiding artefacts. However, they generally lack controllability of the resulting sound texture. After a review and classification of existing approaches, we propose two methods of statistical modeling of the audio descriptors of texture recordings using histograms and Gaussian mixture models. The models can be interpolated to steer the evolution of the sound texture between different target recordings (e.g. from light to heavy rain). Target descriptor values are stochastically drawn from the statistic models by inverse transform sampling to control corpus-based concatenative synthesis for the final sound generation, that can also be controlled interactively by navigation through the descriptor space. To better cover the target descriptor space, we expand the corpus by automatically generating variants of the source sounds with transformations applied, and storing only the resulting descriptors and the transformation parameters in the corpus.

1. INTRODUCTION

The synthesis of sound textures is an important application for cinema, multimedia creation, games and installations. Sound textures are generally understood as sound that is composed of many micro-events, but whose features are stable on a larger time-scale, such as rain, fire, wind, crowd sounds. We must distinguish this from the notion of *soundscape*, which describes the sum of sounds that compose a scene, each component of which could be a sound texture.

The many existing methods for sound texture synthesis are very often concerned with the extension of a given recording to play arbitrarily long, while keeping its overall properties and avoiding artefacts like looping and audible cut points. However, these methods lack *controllability* of the resulting sound texture. Let's pose an example, that we will use throughout the article: A beginning rainfall, that starts with just a few drops, then thickens, until becoming heavy rain. Even if we have several recordings of the different qualities of rain at our disposal, the existing methods couldn't render the gradual evolution of the rain sound.

To achieve this, we propose a method of statistical modeling of the audio descriptors of texture recordings, that

can then be used, varied, or interpolated with other models. Also the steering of the evolution of the generated sound texture is possible, either by giving a target directly in terms of audio descriptors, or deriving these from an existing recording, that couldn't be used directly, e.g. for lack of sound quality or match with the rest of the sound track. Our method is thus strongly based on corpus-based concatenative synthesis (CBCS) [1, 2], which is a new contribution to the field of sound texture synthesis. The use of content-based descriptors is also vastly superior to the often scarce or non-existing meta-data.

CBCS makes it possible to create sound by selecting snippets of a large database of pre-recorded audio (the corpus) by navigating through a space where each snippet is placed according to its sonic character in terms of audio descriptors, which are characteristics extracted from the source sounds such as pitch, loudness, and brilliance, or higher level meta-data attributed to them. This allows one to explore a corpus of sounds interactively or by composing paths in the space, and to create novel timbral evolutions while keeping the fine details of the original sound.

2. RELATED WORK

We will first give an overview of the existing work in sound texture synthesis. As a starting point, Strobl et al. [3] provide an attempt at a definition of sound texture, and an overview of work until 2006. They divide methods into two groups:

Methods from computer graphics Transfer of computer graphics methods for visual texture synthesis applied to sound synthesis [4, 5, 6].

Methods from computer music Synthesis methods from computer music or speech synthesis applied to sound texture synthesis [7, 8, 9, 10, 11].

A newer survey of tools in the larger field of sound design and composition by Misra and Cook [12] follows the same classification as we propose in section 2.1 below. The article makes a point that different classes of sound require different tools (*"A full toolbox means the whole world need not look like a nail!"*).

Filatriau and Arfib [13] review texture synthesis algorithms from the point of view of gesture-controlled instruments, which makes it worthwhile to point out the different usage contexts of sound textures:

There is a possible confusion in the literature about the precise signification of the term *sound texture* that is dependent on the intended usage. We can distinguish two frequently occuring usages:

Expressive free synthesis Here, the aim is to interactively generate sound for music composition, performance, or sound art, very often as an expressive digital musical instrument (DMI, e.g. in [13] and [14]). *Sound texture* is then often meant to distinguish the generated sound material from tonal and percussive sound.

The methods employed for expressive texture generation can give rise to naturally sounding textures, as noted by DiScipio [9], but no systematic research on the usable parameter space has been done, and it is up to the user (or player) to constrain herself to the natural part.

Natural texture resynthesis tries to synthesise textures as part of a larger soundscape. Often, a certain degree of realism is striven for (like in photorealistic texture image rendering), but for most applications, either symbolic or impressionistic *credible texture synthesis* is actually sufficient, in that the textures convey the desired ambience or information, e.g. in simulations for urbanistic planning.

2.1 Classification of Synthesis Methods

It seems most appropriate to divide the different approaches to sound texture generation by the synthesis methods (and analysis methods, if applicable) they employ.

Subtractive and additive synthesis, like noise filtering [10, 11, 15] and additive sinusoidal synthesis [16] are the "classic" synthesis methods for sound textures, often based on specific modeling of the source sounds. [1]

Physical modeling can be applied to sound texture synthesis, with the drawback that a model must be specifically developed for each class of sounds to synthesise (e.g. friction, rolling, machine noise) [5, 17], the latter adding an extraction of the impact impulse sound and a perceptual evaluation of the realism of synthesised rolling sounds.

Granular synthesis uses snippets of an original recording, and possibly a statistical model of the (re)composition of the grains [4, 6, 7, 8, 18, 19].

Corpus-based concatenative synthesis can be seen as a content-based extension of granular synthesis [1, 2]. It is a new approach for sound texture synthesis [20, 21, 22]. Notably, Picard [23] uses grain selection driven by a physics engine.

Non-standard synthesis methods, such as fractal synthesis or chaotic maps, are used most often for expressive texture synthesis [9, 13, 14].

There are first attempts to model the higher-level behaviour of whole soundscapes [24], and by using graphs [25, 26].

---

[1] One venerable attempt is *Practical Synthetic Sound Design* by Andy Farnell at http://obiwannabe.co.uk/tutorials/html/tutorials_main.html.

2.2 Analysis Methods for Sound Textures

Methods that analyse the properties of sound textures are rare, some analyse statistical properties [4, 18, 27, 28], some segment [29] and characterise the source sounds by wavelets [7], and some use adaptive LPC segmentation [30]. Only corpus-based concatenative synthesis methods try to characterise the sonic contents of the source sounds by audio descriptors [1, 2, 20, 21, 31].

3. DESCRIPTOR-BASED SOUND TEXTURE SAMPLING

In order to reproduce a given target sound texture, either with its own sound or by other recordings, we model it by accumulating statistics of its audio descriptor distribution over fixed segments (sizes between 2/3 and 1 second are appropriate, depending on the source sounds).

The descriptors are calculated within the CATART system [21] by a modular analysis framework [32]. The used descriptors are: fundamental frequency, periodicity, loudness, and a number of spectral descriptors: spectral centroid, sharpness, flatness, high- and mid-frequency energy, high-frequency content, first-order autocorrelation coefficient (expressing spectral tilt), and energy. Details on the descriptors used can be found in [33] and [34]. For each segment, the mean value and standard deviation of each time-varying descriptor is stored in the corpus, although for our example of short segments of static rain sound the standard deviation is not informative.

We evaluated two different methods of statistical modeling: histograms (section 3.1) and Gaussian mixture models (section 3.2).

3.1 Histograms

In the histogram method, the individual distributions of the per-segment descriptor values for an input texture are estimated using histograms.

Figure 1 shows the histograms for three classes of rain for 6 descriptors. The corpus is comprised of 2666 units of length 666 ms in 19 sound files of total length of 29.5 minutes from the *SoundIdeas* database, with 701 units for light rain, 981 for medium rain, and 984 for heavy rain. For this corpus, the descriptors are more or less mutually independent, which means that the conceptually simple histogram method gives acceptable results.

For the control of resynthesis, we use the method known as *inverse transform sampling*, where these histograms are interpreted as probability density functions (PDF), from which we calculate the cumulative sum to obtain the CDF (cumulative density function). We then draw random bin indices accordingly by accessing the CDF by a uniformly distributed random value, and draw a uniformly distributed random descriptor value within the bin in order to generate a stream of target descriptor values that obeys the same distribution as the target, in the limits of the discretisation of the histogram.

The resulting distributions can be easily interpolated to generate a smooth evolution from one texture to the next.
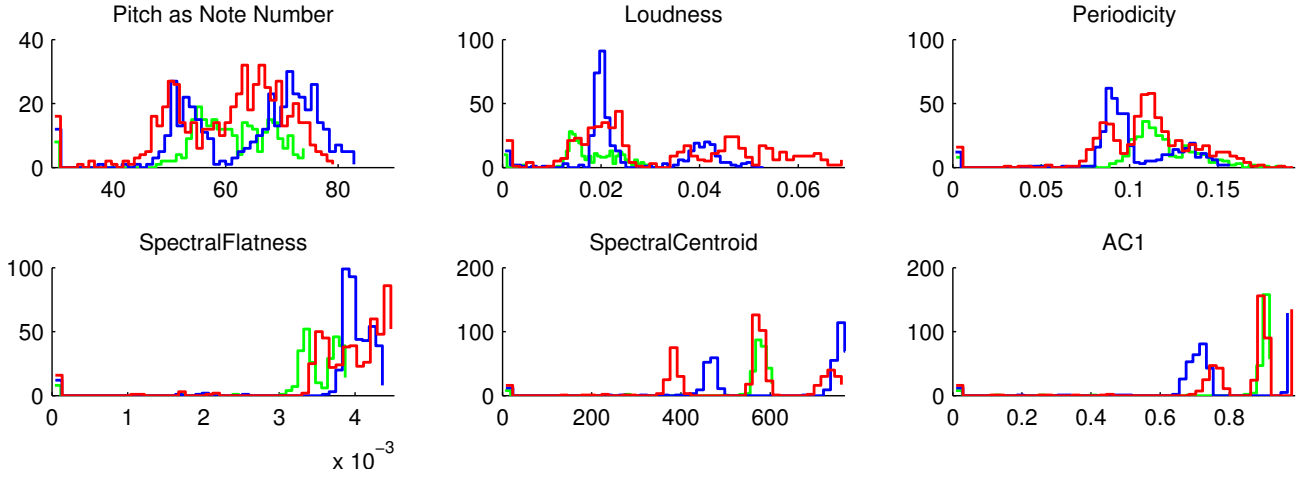
Figure 1. Histograms of spectral centroid, loudness, spectral flatness for the three classes of light (green/clear grey), medium (blue/dark grey), heavy (red/medium grey) rain over a corpus of 2666 segments.

These target descriptors then serve to control a CBCS engine with a corpus of source sounds, as explained in section 3.3.

3.2 Gaussian Mixture Models

In order to capture possible dependencies between the distributions of descriptor values, in this method, we model them by *Gaussian mixture models* (GMMs).

Figure 2 shows the probability density of a two-element mixture for our test corpus, and the interdependencies between two descriptors.

GMMs can be estimated efficiently by Expectation–Maximization. The EM algorithm finds the parameters of a mixture of m multivariate d-dimensional normal distributions:

$$P_j(x|\mu_j, \Sigma_j) = \frac{1}{(2\pi)^{\frac{d}{2}} \det(\Sigma_j)^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)}$$

(1)

where μ_j are the centres, and Σ_j the covariance matrix. Each mixture component is chosen with a *prior* probability of p_j.

For the control of resynthesis, we first choose the component j of the GMM according to the prior probabilities p_j, and then draw values from the component j by taking advantage of the affine transformation property of normal distributions as

$$P_j = \mu_j + A_j \operatorname{erf}(Z)$$

(2)

with Z a uniformly distributed vector, erf the *error function*, i.e. the CDF of a Gaussian, and A_j being the lower triangular matrix from the Cholesky decomposition of Σ_j, i.e. $\Sigma_j = A_j^T A_j$.

GMM parameters can also be interpolated, however, the building of the CDFs for resynthesis is computationally more expensive because of the Cholesky decomposition that needs to be recomputed each time the interpolation changes. Also, care has to be taken to match the m GMM components for the interpolation of μ_j and Σ_j. We chose a greedy matching strategy by closeness of the centres.

3.3 Corpus-Based Concatenative Synthesis

The resynthesis of textures is driven by a vector x of target values for the d used audio descriptors, drawn from the above distributions. Sounds that fulfill these target values are selected from a corpus of source sounds by corpus-based concatenative synthesis, as explained in the following.

The selection of the unit that best matches a given target is performed by evaluating a weighted Euclidean distance C^t that expresses the match between the target x and a database unit u_n with

$$C^t(u_n, x) = \sum_{i=1}^{d} w_i^t C_i^t(u_n, x)$$

(3)

based on the normalized per-descriptor distances C_i^t for descriptor i between target descriptor value $x(i)$ and database descriptor value $u_i(i)$, normalised by the standard deviation σ_i of this descriptor over the corpus:

$$C_i^t(u_n, x) = \left(\frac{x(i) - u_n(i)}{\sigma_i} \right)^2$$

(4)

Either the unit with minimal distance C^t is selected, or one is randomly chosen from the units within a radius r with $C^t < r^2$, or from the set of the k closest units to the target.

The weights w_j were determined interactively for our test corpus, with equal weights for the spectral descriptors, and half weight for pitch and loudness.

Synthesis is performed by possibly transforming the pitch, amplitude, or timbre of the selected units, and then concatenating them with a short overlap, which is sufficient to avoid artefacts for our texture sounds. One additional transformation is the augmentation of the texture density by triggering at a faster rate than given by the units' length, thus layering several units.

Our synthesis engine (see section 4) works in real time, which allows interactive control of the resulting textures. Therefore, and also because we do not model the transitions between units, the unit selection does not need to use sequence-based matching with the Viterbi algorithm [33].
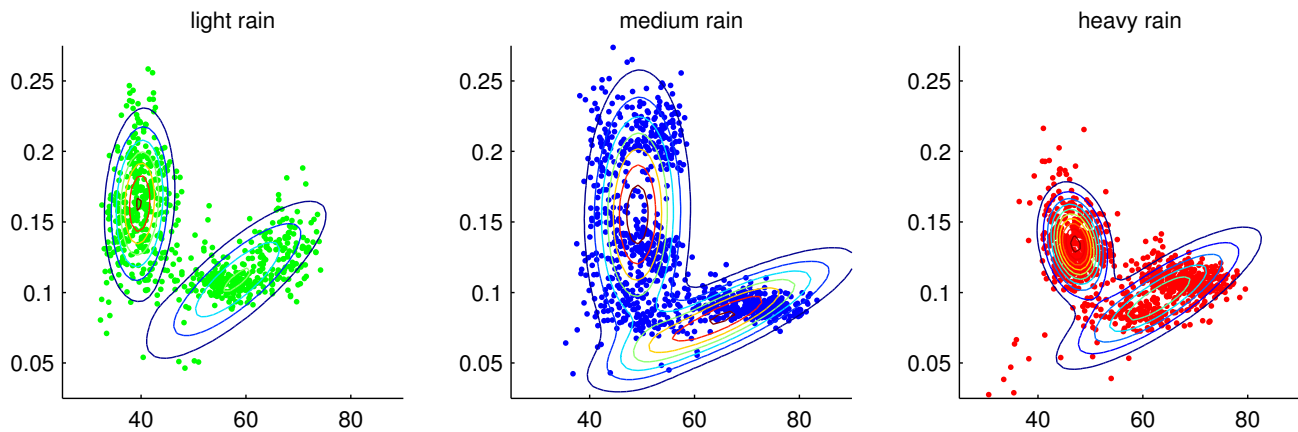
Figure 2. Probability density contours projected on the NoteNumber/Periodicity plane of a Gaussian mixture model of three classes of rain.

3.4 Corpus Expansion

One remaining problem that has not yet been addressed is the possibility that the corpus might not cover the whole range of interpolated and stochastically generated target descriptors. With interactive navigation, we can avoid this shortcoming by judicious tweaking of the playback parameters such as pitch, gain, and filters. In the retargetting case, however, it is hard to derive the necessary transformations from the target values.

This problem could be solved by applying *Feature Modulation Synthesis* (FMS), with the existing research just at its beginning [35]. FMS is concerned with finding the precise sound transformation and its parameters to apply to a given sound, in order to change its descriptor values to match given target descriptors. The difficulty is here that a transformation usually modifies several descriptors at once, e.g. pitch shifting by resampling changes the pitch and the spectral centroid. Recent approaches [36] therefore try to find transformation algorithms that only change one descriptor at a time.

We can get around this problem using a data-driven corpus-based approach, by automatically generating variants of each unit with a certain number and amount of transformations applied, analysing their sound descriptors, and storing only the descriptors and the transformation parameters. The resulting sounds can be easily regenerated on playback.

We generate 5 steps of transpositions by resampling 1 half-tone around the original pitch, and 3 cutoff settings of gentle low-pass and high-pass filters in order to enlarge the timbral variety of the source corpus. The effects of this expansion can be seen in figure 3: a much larger part of the descriptor space between and around the original units is covered by the corpus enlarged 45-fold.

Note that the augmentation of the corpus size does not penalise the runtime of the unit selection much, since we use an efficient kD-tree search algorithm [37] where each doubling of the corpus only adds one more search step on average.
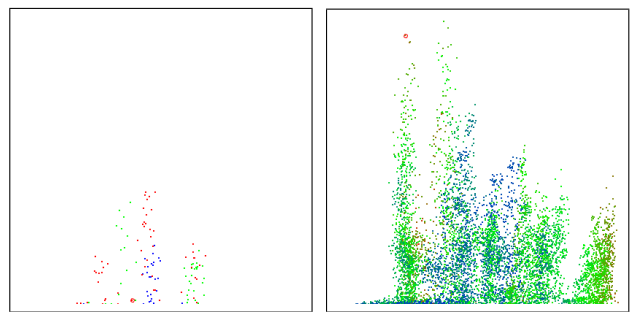

Figure 3. Scatter plot of a texture corpus before (left) and after expansion (right). The x/y/colour axes are spectral centroid, loudness, periodicity.

4. APPLICATIONS AND RESULTS

Our prototype texture synthesiser is implemented in the CATART system [2] [21] for MAX/MSP with the extension libraries FTM&CO [3] [38] making it possible to navigate through a two- or more-dimensional projection of the descriptor space of a sound corpus in real-time, effectively extending granular synthesis by content-based direct access to specific sound characteristics.

The statistical modeling, interpolation, and generation of probability distributions is conveniently handled by the modules mnm.hist, mnm.gmmem, ftm.inter, mnm.pdf from the MnM library [39] included in FTM&CO.

Figure 4 shows an example result using the density parameter, starting from 1 to 10-fold density, resulting in a convincing, albeit quick progression from light rain to a heavy shower. This effect is visible in the gradual whitening of the spectrum. This and other sound examples can be heard on http://demos.concatenative.net.

A creative application of the principle we presented is given in [31], where a musical score for an ensemble was generated from an analysis of sound textures like melting snow or glaciers.

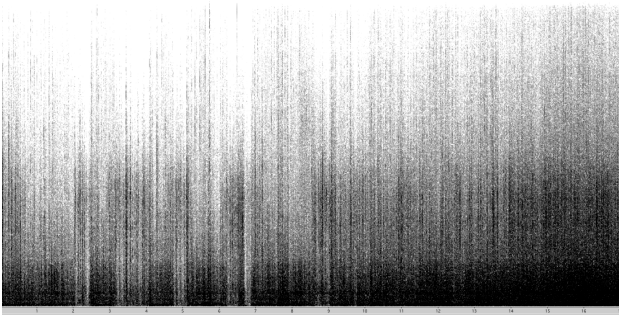
[2] http://imtr.ircam.fr/index.php/CataRT
[3] http://ftm.ircam.fr

Figure 4. Spectrogram of synthetically densifying rain.

5. CONCLUSION AND FUTURE WORK

The sound textures resulting from our descriptor-driven texture synthesis approach using corpus-based concatenative synthesis stay natural whilst being highly controllable. This goes beyond previous approaches that use an existing recording that is extended in time.

We rely on relatively long segments that capture the fine temporal structure of the sounds, and on crossfade and layering to smooth out timbral changes between units. For less overlap, however, abrupt spectral changes can be noticeable, which could be alleviated in two ways in future work: First, we could take into account the timbral transitions in the selection, avoiding too large jumps in the descriptor space. Second, we could apply the grain segmentation approaches described in section 2.2 and work with the unitary micro-events constituting the source textures (for instance, reconstitute rain by grains of water drop length, cut out of the source sounds).

Code is being developed at the moment that adds a third method of statistical modeling by kernel density estimation. The resulting smoothed d-dimensional histogram captures the interdependencies of the descriptors, unlike the separate histogram method in section 3.1, while allowing a more detailed modeling of the descriptor distribution than GMMs in section 3.2.

The brute-force method of corpus expansion (section 3.4) could be easily optimised by applying a greedy strategy that tries to fill only the "holes" in the descriptor space between existing clusters of sounds. Starting from random transformation parameters, if we hit a hole, we'd explore neighbouring parameters until a desired density of the space is reached.

Finally, the biggest restriction to our modeling approach lies in the assumption of stationarity of the source textures. This is appropriate for many interesting textures, but already rain with intermittent thunder sounds wouldn't be modeled correctly. Clearly, clustering and modeling of the transitions between clusters using hidden Markov models (HMMs) or semi-Markov models seems promising here. This would base the graph approach introduced in [25] on actual data, and could also model the larger-scale temporality of sound scapes as a sequence of textures.

7. REFERENCES

[1] D. Schwarz, "Concatenative sound synthesis: The early years," *Journal of New Music Research*, vol. 35, pp. 3–22, Mar. 2006. Special Issue on Audio Mosaicing.

[2] D. Schwarz, "Corpus-based concatenative synthesis," *IEEE Signal Processing Magazine*, vol. 24, pp. 92–104, Mar. 2007. Special Section: Signal Processing for Sound Synthesis.

[3] G. Strobl, G. Eckel, D. Rocchesso, and S. le Grazie, "Sound texture modeling: A survey," in *Proceedings of the Sound and Music Computing Conference*, 2006.

[4] S. Dubnov, Z. Bar-Joseph, R. El-Yaniv, D. Lischinski, and M. Werman, "Synthesis of audio sound textures by learning and resampling of wavelet trees," *IEEE Computer Graphics and Applications*, vol. 22, no. 4, pp. 38–48, 2002.

[5] J. O'Brien, C. Shen, and C. Gatchalian, "Synthesizing sounds from rigid-body simulations," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 175–181, ACM New York, NY, USA, 2002.

[6] J. Parker and B. Behm, "Creating audio textures by example: tiling and stitching," *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, vol. 4, pp. iv–317–iv–320 vol.4, May 2004.

[7] R. Hoskinson and D. Pai, "Manipulation and resynthesis with natural grains," in *Proceedings of the International Computer Music Conference (ICMC)*, (Havana, Cuba), pp. 338–341, Sept. 2001.

[8] C. Bascou and L. Pottier, "GMU, A Flexible Granular Synthesis Environment in Max/MSP," in *Proceedings of the Sound and Music Computing Conference*, Citeseer, 2005.

[9] A. Di Scipio, "Synthesis of environmental sound textures by iterated nonlinear functions," in *Digital Audio Effects (DAFx)*, 1999.

[10] M. Athineos and D. Ellis, "Sound texture modelling with linear prediction in both time and frequency domains," *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, vol. 5, pp. V–648–51 vol.5, April 2003.

[11] X. Zhu and L. Wyse, "Sound texture modeling and time-frequency LPC," in *Digital Audio Effects (DAFx)*, vol. 4, 2004.

[12] A. Misra and P. Cook, "Toward synthesized environments: A survey of analysis and synthesis methods for sound designers and composers," in *Proc. ICMC*, 2009.

[13] J. Filatriau and D. Arfib, "Instrumental gestures and sonic textures," in *Proceedings of the International Conference on Sound and Music Computing (SMC)*, 2005.

[14] J. Filatriau, D. Arfib, and J. Couturier, "Using visual textures for sonic textures production and control," in *Digital Audio Effects (DAFx)*, 2006.

[15] M. Lagrange, B. Giordano, P. Depalle, and S. McAdams, "Objective quality measurement of the excitation of impact sounds in a source/filter model," *Acoustical Society of America Journal*, vol. 123, p. 3746, 2008.

[16] S. Guidati and Head Acoustics GmbH, "Auralisation and psychoacoustic evaluation of traffic noise scenarios," *Journal of the Acoustical Society of America*, vol. 123, no. 5, p. 3027, 2008.

[17] E. Murphy, M. Lagrange, G. Scavone, P. Depalle, and C. Guastavino, "Perceptual Evaluation of a Real-time Synthesis Technique for Rolling Sounds," in *Conference on Enactive Interfaces*, (Pisa, Italy), 2008.

[18] Z. El-Yaniv, D. Werman, and S. Dubnov, "Granular Synthesis of Sound Textures using Statistical Learning," in *Proc. ICMC*, 1999.

[19] M. Fröjd and A. Horner, "Sound texture synthesis using an overlap-add/granular synthesis approach," *Journal of the Audio Engineering Society*, vol. 57, no. 1/2, pp. 29–37, 2009.

[20] D. Schwarz, G. Beller, B. Verbrugghe, and S. Britton, "Real-Time Corpus-Based Concatenative Synthesis with CataRT," in *Digital Audio Effects (DAFx)*, (Montreal, Canada), Sept. 2006.

[21] D. Schwarz, R. Cahen, and S. Britton, "Principles and applications of interactive corpus-based concatenative synthesis," in *JIM*, (GMEA, Albi, France), Mar. 2008.

[22] M. Cardle, "Automated Sound Editing," tech. rep., Computer Laboratory, University of Cambridge, UK, May 2004.

[23] C. Picard, N. Tsingos, and F. Faure, "Retargetting Example Sounds to Interactive Physics-Driven Animations," in *n AES 35th International Conference, Audio in Games.*, (London Royaume-Uni), 2009.

[24] D. Birchfield, N. Mattar, and H. Sundaram, "Design of a generative model for soundscape creation," in *International Computer Music Conference, Barcelona, Spain*, Citeseer, 2005.

[25] A. Valle, V. Lombardo, and M. Schirosa, "A graph-based system for the dynamic generation of soundscapes," in *Proceedings of the 15th International Conference on Auditory Display*, (Copenhagen), pp. 217–224, 18–21 May 2009.

[26] N. Finney, "Autonomous generation of soundscapes using unstructured sound databases," Master's thesis, MTG, IUA–UPF, Barcelona, Spain, 2009.

[27] M. Desainte-Catherine and P. Hanna, "Statistical approach for sound modeling," in *Digital Audio Effects (DAFx)*, Citeseer, 2000.

[28] C. Bascou and L. Pottier, "New sound decomposition method applied to Granular Synthesis," in *Proc. ICMC*, (Barcelona, Spain), 2005.

[29] S. O'Modhrain and G. Essl, "PebbleBox and CrumbleBag: tactile interfaces for granular synthesis," in *New Interfaces for Musical Expression*, (Singapore), 2004.

[30] I. Kauppinen and K. Roth, "An Adaptive Technique for Modeling Audio Signals," in *Digital Audio Effects (DAFx)*, Citeseer, 2001.

[31] A. Einbond, D. Schwarz, and J. Bresson, "Corpus-based transcription as an approach to the compositional control of timbre," in *Proc. ICMC*, (Montreal, QC, Canada), 2009.

[32] D. Schwarz and N. Schnell, "A modular sound descriptor analysis framework for relaxed-real-time applications," in *Proc. ICMC*, (New York, NY), 2010.

[33] D. Schwarz, *Data-Driven Concatenative Sound Synthesis*. Thèse de doctorat, Université Paris 6 – Pierre et Marie Curie, Paris, 2004.

[34] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the Cuidado project," Tech. Rep. version 1.0, Ircam – Centre Pompidou, Paris, France, Apr. 2004.

[35] M. Hoffman and P. Cook, "Feature-based synthesis: mapping acoustic and perceptual features onto synthesis parameters," in *Proc. ICMC*, (Copenhagen, Denmark), 2006.

[36] T. Park, J. Biguenet, Z. Li, C. Richardson, and T. Scharr, "Feature modulation synthesis (FMS)," in *Proc. ICMC*, (Copenhagen, Denmark), 2007.

[37] D. Schwarz, N. Schnell, and S. Gulluni, "Scalability in content-based navigation of sound databases," in *Proc. ICMC*, (Montreal, QC, Canada), 2009.

[38] N. Schnell, R. Borghesi, D. Schwarz, F. Bevilacqua, and R. Müller, "FTM—Complex Data Structures for Max," in *Proc. ICMC*, (Barcelona), 2005.

[39] F. Bevilacqua, R. Muller, and N. Schnell, "MnM: a Max/MSP mapping toolbox," in *New Interfaces for Musical Expression*, (Vancouver), pp. 85–88, 2005.