SOUND-TYPES: A NEW FRAMEWORK FOR SYMBOLIC SOUND ANALYSIS AND SYNTHESIS

Carmine Emanuele Cella

Università di Bologna
Via Zamboni 33 - 40126 Bologna - Italy
carmine.emanuele.cella@gmail.com

ABSTRACT

Sound-types are a new method to represent and manipulate sounds in a quasi-symbolic way by means of low-level features and subsequent analysis stages. After the presentation of the basic ideas, a full analysis-synthesis framework and some applications will be shown.

1. INTRODUCTION

A typical way to represent musical signals is a decomposition of a time-sequence $x[n]$ into a linear combination of the form:

$$x[n] = \sum_{k=1}^{K} \alpha_k g_k[n]. \quad (1)$$

The coefficients α_k are derived from an analysis stage, while the functions $g_k[n]$ can or cannot be determined by the analysis stage and are used during a synthesis stage; both stages are related to a particular signal model [6], [5].

Equation 1 is the so-called *signal-level* representation; this is rather general, computationally efficient, invertible[1] and expresses some physical properties associated to the signal. However, more expressive representaions, are also possible.

Symbolic-level representations[2] can be used to express complex relationships and hierarchies by means of symbols that encode musical information (structure analysis, repeated patterns, etc.) but are usually inefficient, non-invertible and are hardly related to the physical nature of sound (figure 1) [12].

Mid-level representations, finally, try to address the issue related to the lack of expressivity by focusing on *relatively simple* concepts that are, however, more abstract than the basis of the analysis used in equation 1. These concepts are usually based on perceptual criteria related to the low-level hearing and are situated in between the constraints imposed on them by lower and higher levels [4].

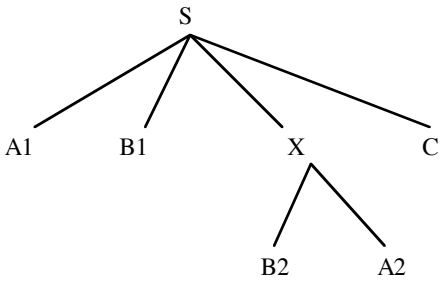

Figure 1. An example of symbolic-level representation; here the symbols represent musical entities hierarchically organized.

All the representation levels discussed so far have a *fixed degree of abstraction*. In other words, they focus on a particular point of view and are not scalable: once a representation level has been selected it is not possible to go smoothly to another level. All of them impose *their own* concepts onto the signal: each representation models the signal with it's own concepts, even if they are completely irrelevant to *that* particular signal; figure 2 roughly depicts the described ideas.

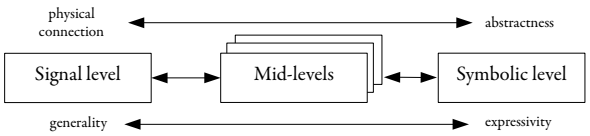

Figure 2. The levels of representation.

The main purpose of this article is to propose a connection between the signal and the symbolic-level by defining a new representation method based on specific signal processing techniques able to retrieve information from a signal and model that information statistically to find *salient* properties. After a theoretical formulation of the proposed representation method, a real implementation will be presented, showing how this method is a full new framework for sound analysis and synthesis.

[1] With *invertibility* we mean the possibility to go back to the signal domain from the representation itself.

[2] Historically, symbolic-level representations designed *highly formalized descriptions of music*, possibily based on a formal language and on its underlying logic [1]. First attempts to apply formal logic to music rely mainly on a deductive system called *first-order logic*. Later on, inspired by linguistic ideas, other extensions of logic have also been tested (temporal, modal, non-monotonic, etc.) [8], [9].

2. SOUND-TYPES

With *sound-types* we define a new representation method for musical signals that, while being generic enough to be used for different signals, fulfills *by-design* the following requirements:

- **signal-dependent semantics**: the basis of the representation are inferred from the signal, using learning techniques; this creates the possibility to describe concepts that are really *related* to the sound being analysed (adaptive dictionary);

- **scalability**: it is possible to change the *degree of abstraction* in the representation, ranging from the signal level to the symbolic- level in a *continuous* manner; the degree of abstraction becomes a parameter of the representation;

- **weak invertibility**: the representation method is able to generate the represented signal; this possibility does not imply, however, that the generated signal must be waveform-identical to the original one, but only that *perceptually relevant* parts of it can be reconstructed (that's why we call it weak);

- **generativity**: it is possibile to generate sounds *other* than the original one, according to some parameters in the domain of the representation that can be estimated from a given signal or deliberately created.

2.1. The *typed* model

The basic idea of sound-types is to represent sounds by means of *types* and *rules* inferred by some low-level descriptions of signals [10] and subsequent learning stages. The types represent **classes of equivalences** for sounds, while the rules represent **transition probabilities** that a type is followed by another type.

Mathematically, we want to be able to *translate* a signal-level representation into other forms involving different elements and operators (mid to symbolic-level representations); more formally:

$$\begin{aligned} x[n] &= \textstyle\sum_{k=1}^{K} \alpha_k g_k[n] \\ &= \alpha_1 g_1[n] + \ldots + \alpha_k g_k[n] \\ &= \beta_1 f_1[n] + \ldots + \beta_j f_j[n] \\ &\vdots \\ &= \omega_1 h_1[n] + \ldots + \omega_t h_t[n]. \end{aligned}$$

In the equations above $\alpha, \beta, \ldots, \omega$ could be any kind of weighting coefficients, $g_k, f_j, \ldots, h_t$ are variables belonging to different *types*, $+$ and $\cdot$ are relations defined for each type and $t < j < \ldots < k$ (i.e. last equation has less elements than first equation). Notice that $+$ and $\cdot$ are not algebraical sum and multiplication and are *not* required to be commutative: they can be any kind of binary relation defined over specific types. As long as it is possible to convert, say, from type g_k to type h_t and to define relations on both we can perform the translation. Since the $+$ relation is not the algebraical sum, we will suppose that our symbolic-level representation is a *sequence* of types and that $+$ is the *successor* function (i.e. $g+f$ means that variable f of type F follows variable g of type G[3]; remember that if F and G are types then $G \rightarrow_+ F$ is a type).

To validate the proposed translation we need some functions that clearly define whether a given variable belongs to a given type and how it is possible to convert from a type to another. A possible way to achieve these requirements is by means of clusters of low-level descriptors in the feature space.

2.2. The analysis stage

In order to provide a verifiable model for the proposed theory, a twofold process divided into the following stages is needed:

- **types inference**: during this stage the types involved in the representations are discovered;

- **rules inference**: a second stage is needed to discover the relations between the types.

This is an iterative process and must be repeated until there are no more rules to discover; this will be cleared later on.

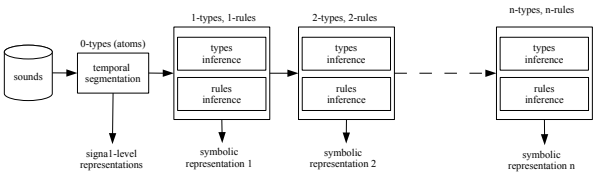

Figure 3. An outline of the proposed algorithm for types and rules inference

The following procedure shows a possible implementation of the twofold process, using low-level descriptors plus statistical learning for types inference and Markov models for rules inference; the first step of the algorithms is represented by temporal segmentation:

1. **atoms creation**: subdivide a sound into small chunks of approximately 40 ms called *atoms* or *0-types* overlapping in time and frequency (labelled them with integer numbers); these atoms can be produced either by simply overlapped windows, by onsets separations or by other approaches such as atomic decomposition;

[3] The *successor* relation is evidently non-commutative.

2. **1-types inference**: compute a set of low-level descriptors on each atom obtained in the previous step, project the descriptors in a multi-dimensional space and compute the *clusters* by means of statistical techniques; each cluster will represent a *1-type* (let's label them $g_1, f_1, \ldots$);

3. **1-rules inference**: implement a Markov model to describe the sequences of types present in the analysed sound (*1-rules)*;

4. **1-level representation**: represent the sound in a symbolic language using the discovered 1-types and 1-rules and create sequences of types depending on the rules;

5. **n-types inference**: compute a set of low-level descriptors on the whole sequences found in previous steps (for example $g_1 + f_1$); project again the descriptors and compute again the clusters: each cluster will represent a *n-type* (let's label them $g_n, f_n, \ldots$);

6. **n-rules inference**: implement a Markov model to describe the sequences of types present in the analysed sound (*n-rules)*;

7. **n-level representation**: represent the sound in a symbolic language using the discovered n-types and n-rules and create sequences of types depending on the rules;

8. **repeat n-rules and r-types**: until valid rules are found.

Algorithm 2 details the described procedure in pseudo-code.

Algorithm 1 Sound-types analysis

Require: signal s
  decompose s in atoms $a[n]$
  repeat
    for every atom in $a[n]$ **do**
      compute m-dimensional feature space $f_{n,m}$
    end for
    compute optimal number of clusters k
    compute clusters $c[k]$ on $f_{n,m}$
    synthesize k types from $c[k]$
    create a representation r of s using clusters $c[k]$
    for every cluster in $c[k]$ **do**
      compute transition probabilities $p_{k,k}$
    end for
    synthesize sequences of types with non-null probability
    $a[n] \Leftarrow$ synthesized sequences of types
    $n \Leftarrow k$
  until no more transitions

The number of iterations of the whole process are called the *abstraction levels* of the representation. In terms of atomic decomposition, all the sets of the discovered types are time-frequency atoms with different time scales and spectral content; the higher the level of a type the less it is generic, the more expressive. Figure 3 illustrates the proposed approach. Low-level descriptors and statistical techniques are not used to classify different sounds, but *parts* of a single sound; another approach could take into account a real population of sounds and compute sound-types over a whole database; since different atoms and sequences (moleculae) belong to the same type as long as they share common properties (defined by the set of descriptors), they could theoretically be shared between different sounds. From an acoustical point of view, the information amount increases dramatically from level to level, ranging from the so-called *acoustical quanta* to segments of sounds that could be even recognized as *sections* of a musical composition. The representaion created on each level can be done on a symbolic language of choice, even with simple strings of labels.

2.3. The synthesis stage

The synthesis of the discovered types is a relatively easy task and can be done either in time of frequency. In time, it is basically a weighted sum of all the atoms belonging to the same cluster, in which the distance of the cluster is the weight. Algorithm 2 details the synthesis procedure in pseudo-code.

Algorithm 2 Sound-types synthesis

Require: n-level representation r
Require: dictionary of n-types $a[n]$
  for every symbol in r **do**
    overlap-add corresponding type from $a[n]$
  end for

Other methods are also possible instead of weighed sum. For example, a *witness* of a type can be selected from the cluster (either randomly or by its distance to the center of the cluster). The overall quality of the reconstructed signal strictly depends on the number of types used and on the synthesis method selected.

3. CURRENT STATUS

Current implementation does not cover all the parts of the analysis-synthesis algorithm for sound-types analysis. A partial implementation can compute low-level features, clusters in the feature space and transition probabilies up to the first level. The analysis-synthesis framework is perfectly functional but the symbolic representation is only possibile with 1-types and 1-rules thus proving the sound-type theory only partially.

A typical types-inference stage (performed by clustering) is represented in figure 4: common sound-atoms are grouped in the same cluster and relevant elements of the clusters (such as centroid, spread, etc.) are computed.

A rules-inference stage (performed by a Markov chain), is depicted in figure 5: the nodes are the sound-types, while the connections are the transitions between them.

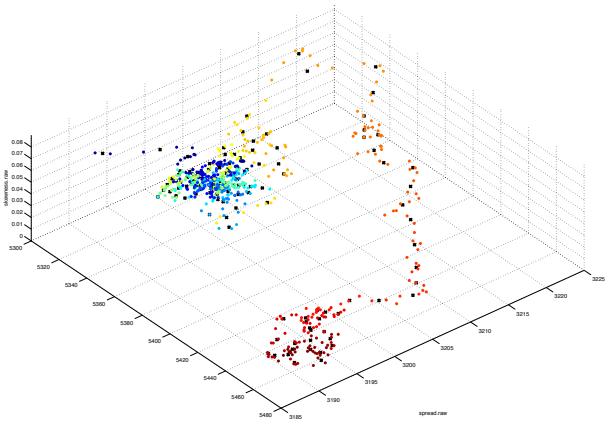

Figure 4. A typical clustering stage.

With the two stages computed for the first level (1-types, 1-rules), it is possible to represent a signal in a pseudo-symbolic way through a *string of labels*. After the analysis stage, a dictionary of the found types (basically sound grains created as described in section 2.3) and and a simple string are produced: each label at position k in the string represents the corresponding type (through a numeric index that refers to the position in the dictionary). In general, the algorithm creates a *compact* representation of the given sound; the size of the representation is directly connected to the number of types discovered. If we represent a sound with 33% of sound-types (i.e. a type each three atoms) the compression ratio will be roughly 60%.

The complete list of implemented features in the current version is the following:

- **low-level features**: spectral centroid, spectral spread, spectral skewness, spectral kurtosis, spectral irregularity, specrtal slope, spectral decrease, high frequency content, spectral flux, energy, zero-crossing rate, fundamental frequency, inharmonicity;
- **clustering algorithms**: k-means, gaussian mixture models (GMM);
- **auto-estimation of number of clusters**: it is possible to automatically estimate the optimal number of clusters by means of two distinct techniques for each clustering algorithm (gap-statistic for k-means and BIC measure for GMM);
- **dimensionality reduction**: by means of principal component analysis (PCA) it is possible to compute as many features as wanted and then reduce the analysis to a smaller number of dimensions;
- **transition probabilities**: Markov chain;
- **resynthesis algorithm**: the resynthesis algorithm works both in frequency and in time domain with types interpolation. In the reconstructed signal, where a long sequence of the same type is found, it is possible to create linear interpolation with next appearing type, improving sound quality;
- **distance measures for resynthesis**: euclidean, Manhattan distance (taxicab), Mahalanobis distance, cosine similarity;
- **symbolic representations**: strings of labels.

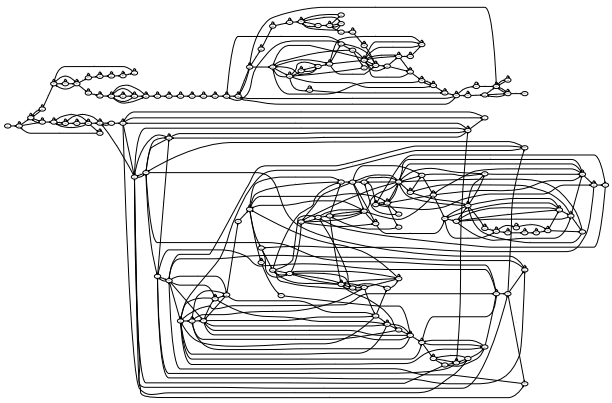

Figure 5. Transitions probabilities for the first level.

3.1. Applications

Applications of the algorithm currently possible are:

- **time and frequency transformations**: it is possible to perform various transformations such as time-stretch and pitch-shift; for the latter, formant preservation by means of cepstral envelope has been also implemented thus creating a sort of advanced phase-vocoder[4].
- **audio compression**: while not being the main purpose of the approach, it is possible to compress a sound by a given ratio. The quality of compression, anyway, is not comparable with dedicated algorithms such as MP3.
- **probabilistic generation**: using the discovered probabilities and types it is possible generated sounds *affine* to the original ones, by means of a biased random generator; we tested this approach on a small jazz corpus (including several instruments) to imitate the style of improvisations;
- **symbolic description**: analysing the created string of labels, it is possible to acquire information of *salient* properties of the sound and represent such information in a meaningful way; figure 6, for example, shows a comparison between a circular graph created with collected types and probabilities (using

[4]To improve the sound quality of the phase-vocoder, the Laroche-Dolson approach [7] has also been implemented.

the same approach as figure 5) and a typical structure representation from the software *OpenMusic*[5].

Some samples processed by the proposed method can be found online at `http://www.soundtypes.com`.

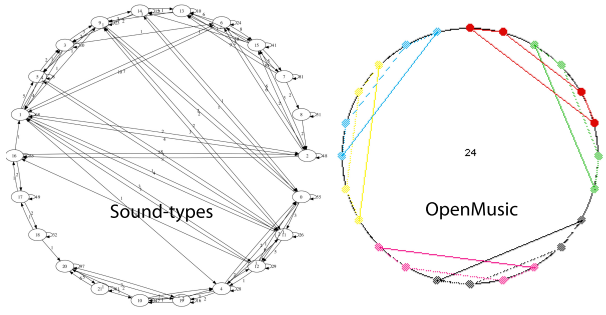

Figure 6. A comparison between sound-types and OpenMusic graphs

4. A GENERALIZED FRAMEWORK

It's important to point out that the proposed algorithm is only a *possible* realization of the general idea (see [2]). The low-level features to be used can be many more and so can be the clustering techniques. Other concatenation methods are also possible (see [3], [11]) and, finally, the symbolic language for the sound representation is a matter of choice. The more expressive the language, the more the possible manipulations on the symbolic-level.

The whole problem of representing signals in symbolic ways is built of three major parts: a **back-end**, a **concatenation layer** and a **front-end**.

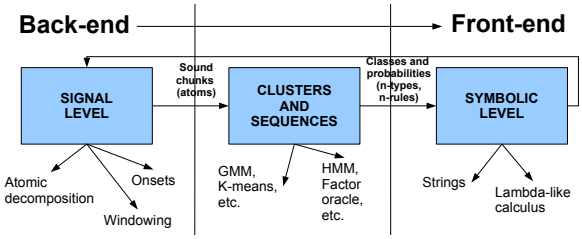

Figure 7. A global framework for the sound-type theory.

The main aim of the back-end is to provide chunks of sounds (atoms) to be subsequently analysed; this can be done simply by windowing or by using more complex techniques such as atomic decomposition or onsets separation. Once atoms have been defined, it's possible to look for *classes of equivalence* for sound and *concatenation* rules between classes in order to decompose a signal; this should be done in the second layer. Once classes and rules have been collected it's then possibile to represent them through a *grammar* of any symbolic language, either descriptive only or generative[6]; this is, finally, the aim of the front-end.

For each level there could be plenty of possibilities in terms of algorithms and techniques; it's therefore mandatory to define a sort of *interface* between levels, in order to have a modular system into which plug different tools on demand. Figure 7 depicts the described ideas.

In the context of this generalized framework, table 1 summarize the implemented features.

Table 1. Implemented techniques for each layer.

LAYER	TECHINQUES
Back-end	Windowing, onsets separation
Concatenation layer	Low-level features + GMM
	Low-level features + K-means
	Markov chains
Front-end	Descriptive language (strings)

Obviously, one of the main lacking features in the current implementation is a powerful symbolic language. Representing signals with a sequence of labels is not enough to permit advanced manipulations in the symbolic-level. In the next sections we will try to summarize the situation and we will point out some possible improvements.

5. CONCLUSIONS AND PERSPECTIVES

The **theory of sound-types** needs expansions and improvements both in symbolic-level and in the signal-processing level. Sound-types seem to be promising entities to represent music because they are physically related to sound, are invertible and are also capable to represent formal relationships and hierarchies.

5.1. Open problems

Many problems are still open; one of the most important is what we call the **reduction effect**: the more we clusterize (meaning that we reduce the number of clusters, grouping more entities in the same sound-type) the better will be the representation but the worst will be the sound resynthesis. Sound quality is directly linked to the number of clusters, but if we augment this number we loose the possibility of having many levels in the analysis. No easy solutions have been found at the moment for this effect.

The theory of sound-types is a full analysis-synthesis framework like, say, the phase-vocoder. Nevertheless, we still lack a comprehensive **mathematical formulation** for the described method that handles correctly types and rules. We can define a sound type as a weighted sum of atoms:

$$t[n] = \sum_{k=1}^{K} \omega_k a_k[n] \qquad (2)$$

[5]This is software is a well known tool developed at IRCAM to perform computer-aided composition; for more information see `http://repmus.ircam.fr/openmusic/home`.

[6]With the word *grammar*, here, we simply mean a corpus of syntactic rules that define any formal system.

where ω_k are the distances from the center of the reference cluster and $a_k[n]$ are the atoms belonging to the reference cluster. We still don't know, anyway, how to deal with transition probabilites for higher levels: the theory, at the moment, only looks for *non-null* probabilities without giving relevance to more probable sequences of sound-types.

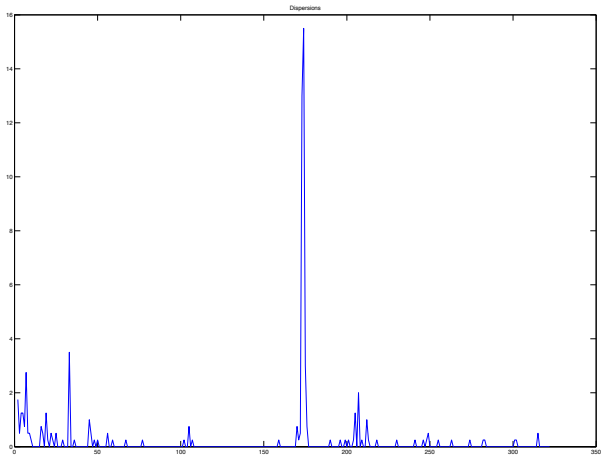

Figure 8. A plot of clusters dispersions as a quality measure of the method; the highest peaks signify bad clusters.

5.2. Evaluation

It is not easy fo find **evaluation procedures** for the proposed method. The first technique the we adopted is acoustic inspection of reconstructed signals after full analysis and synthesis. Afterward we introced a measure of *clustering quality* by means of the gap statistic. Using that measure we are able to evaluate bad clustering looking at dispersions: in figure 8 the highest peaks signify bad clusters. Anyway, this quality measure is hardly related to the quality of the reconstructed signal and a good evaluation procedure is still under investigation.

5.3. Future work

The next important steps in the research will be the following:

- **typed language**: the representation on the symbolic-level is now a simple string while it should be done in a appropriate language that can handle types; for this reason an investigation in simply-typed languages will be done and a more expressive language for symbolic representation will be supported;
- **higher levels**: higher levels are still not implemented; this could put into the game the transition probabilites and lead to a complete mathematical formulation of the theroy;
- **symbolic-level transformations**: it should be possible to transform a sound working on the string of labels. For example, pitch-shift or time-stretch only selected types or extract some given types to perform semantic source separation.

When some of these imporant steps will be complete, it will be also possible to propose more appropriated evaluation techniques in order to understand the real power of the p-roposed approach.

6. REFERENCES

[1] C. E. Cella, "Sulla struttura logica della musica," in *Rivista umbra di musicologia*, vol. vol. 48, 2004, pp. pp. 1–82.

[2] ——, "Towards a symbolic approach to sound analysis," in *Mathematics and Computation for Music Conference (MCM)*, Yale University - New Haven, 2009.

[3] A. Cont, "Audio oracle: A new algorithm for fast learning of audio structures," in *International Computer Music Conference (ICMC)*, 2007.

[4] D. Ellis and D. Rosenthal, "Mid-level representations for computational auditory scene analysis," in *International joint conference on Artificial Intelligence*, 1995.

[5] M. Goodwin, *Adaptive signal models*. Kluwer Academic Publishers, 1998.

[6] M. Goodwin and M. Vetterli, "Atomic decompositions of audio signal," in *IEEE Audio Signal Processing Workshop*, 1997.

[7] J. Laroche and M. Dolson, "New phase-vocoder techniques for real-time pitch shifting, chorusing, harmonizing, and other exotic audio modifications," in *Journal of the Audio Engineering Society*, vol. vol. 47, 1999, pp. pp. 928–936.

[8] M. Leman, "Expressing coherence of musical perception in formal logic," in *Mathematics and music: a Diderot Mathematical Forum*, 2002, pp. pp. 184–198.

[9] A. Marsden, "Timing in music and modal temporal logic," in *Journal of Mathematics and Music*, vol. vol. 1, No. 3, 2007, pp. pp. 173–189.

[10] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," in *CUIDADO I.S.T. Project Report*, April 2004, pp. 1–25.

[11] ——, "Sequence representation of music structure using higher-order similarity matrix and maximum-likelihood approach," in *ISMIR Wien*, 2007.

[12] H. Vinet, "The representation levels of music information," in *Lecture Notes in Computer Science,Springer Verlag*, vol. vol. 2771, 2003.