General Sound Classification and Similarity in MPEG-7

Michael Casey, *MERL Cambridge Research Laboratory*

***Abstract*— we introduce a system for generalized sound classification and similarity using a machine learning framework. Applications of the system include automatic classification of environmental sounds, musical instruments, music genre and human speakers. In addition to classification, the system may also be used for computing similarity metrics between a target sound and other sounds in a database. We discuss the use of hidden Markov models for representing the temporal evolution of audio spectra and present results of testing the system on classification and retrieval tasks. The system has been incorporated into the MPEG-7 international standard for multimedia content description and is therefore publicly available in the form of a set of standardized interfaces and software reference tools for developers and researchers.**

Keywords: de-correlated features, sound classification, sound similarity, generalized timbre, finite-sate model.

I. INTRODUCTION

As databases of sound samples and music files become larger and more interconnected, composers and sound designers are faced with new challenges in content management. In this paper, we address the problem of general sound organization by machines in order to support applications in sound archiving, classification and retrieval. General sound is defined loosely to be any audio clip with no assumptions on length, segmentation, source category or composition with other sounds; such as mixtures and textures. Examples of classification include recognizing the category of a sound such as speech, non-speech vocal utterance, environmental, musical, silence and various mixed auditory scenes. At a finer level of detail classification can pinpoint sources such as clarinet, crickets, or *Michael Casey*. Retrieval by similarity requires automatic ordering of audio segments to construct a graded sequence of sonic materials sorted by distance from a target sound.

MPEG-7 is a new international standard for media content description that is well suited to applications in music indexing, similarity matching, and knowledge-based audio processing, see (International Standards Organization 2001). Support for content management applications was a one of the design requirements for the standard and, as such, it consists of methods for computing feature extraction, similarity, and classification for a wide range of audio and visual media. By introducing these methods in the public domain, MPEG-7 will likely have a similar scale of impact on the future of music technology as the MIDI and MPEG layer III audio (MP3) standards have had in the past.

The MPEG-7 general sound recognition tools use de-correlated spectral features coupled with hidden Markov models (HMM) for computing similarity and generating source classifications. In addition to traditional timbre methods that apply only to isolated musical instrument notes, MPEG-7 also represents noise textures, environmental sounds, music recordings, melodic sequences, vocal utterances and sounds containing mixtures of sources. For some recent work in the area of sound indexing and retrieval, see (Wold, Blum, Keislar, and Wheaton 1996; Boreczky and Wilcox 1998; Martin and Kim1998; Zhang and Kuo 1998).

In this paper we shall discuss two new novel components of MPEG-7 audio. The first is the use of de-correlated spectral features for low-dimensional sound representation. The second component is source identification and general sound similarity using finite-state probabilistic inference models, see (Casey and Westner 2000; Casey 2001a; Casey 2001b; Casey 2001c). These methods were chosen from a range of competing technologies and were found to exhibit good performance in a variety of applications. The tools will likely be useful to composers for automatically organizing sonic materials using computational methods. The tools will also be of interest to software developers and researchers for building new advanced applications in music and audio processing.

II. MPEG-7 Content Description Interface

The MPEG-7 standard consists of descriptors and description schemes that are defined using a modified version of XML schema called the *description definition language* (DDL). A large number of descriptors have been defined covering images, audio, video and general multimedia usage. The DDL language ensures that media content description data may be shared between applications in much the same way that sound files are exchanged using standard file formats. For example, an audio spectrum is defined by a descriptor called `AudioSpectrumEnvelope.` To use the descriptor, data is instantiated using the standardized DDL syntax. In this case, the spectrum data is stored as a series of vectors within the class, see DDL Example 1.

```
<!-- ################################################################ -->
<!-- Definition of audioSpectrumAttributeGrp                        -->
<!-- ################################################################ -->
<attributeGroup name="audioSpectrumAttributeGrp">
  <annotation>
    <documentation>Edge values are in Hertz</documentation>
  </annotation>
  <attribute name="loEdge" type="float" use="default" value="62.5"/>
  <attribute name="hiEdge" type="float" use="default" value="16000"/>
  <attribute name="resolution">
    <simpleType>
      <restriction base="string">
        <enumeration value="1/16 octave"/>
        <enumeration value="1/8 octave"/>
        <enumeration value="1/4 octave"/>
        <enumeration value="1/2 octave"/>
        <enumeration value="1 octave"/>
        <enumeration value="2 octave"/>
        <enumeration value="4 octave"/>
        <enumeration value="8 octave"/>
      </restriction>
    </simpleType>
  </attribute>
</attributeGroup>

<!-- ################################################################ -->
<!-- Definition of AudioSpectrumEnvelopeType                        -->
<!-- ################################################################ -->
<complexType name="AudioSpectrumEnvelopeType">
  <complexContent>
    <extension base="mpeg7:AudioLLDVectorType">
      <attributeGroup ref="mpeg7:audioSpectrumAttributeGrp"/>
    </extension>
  </complexContent>
</complexType>
```

DDL Example 1. Standardized XML-Schema definition of an audio spectrum description.

This descriptor enumerates allowable values for bands of logarithmically spaced power spectra. The spectrum values are contained in an encapsulated descriptor called `AudioLLDVectorType,` read *audio low-level descriptor vector type*, that is defined as a series of floating point numbers. The process for extracting values for the power spectrum is outside the scope of the standard, but guidelines are provided in order that implementers of the standard conform to certain constraints thus ensuring interoperability of descriptions generated by different implementations.

Figure 1 shows the distribution of power spectrum coefficients for a one octave bandwidth power spectrum. In addition to the eight in-band coefficients, two additional coefficients enumerate the total power below and above the within-band limits respectively.

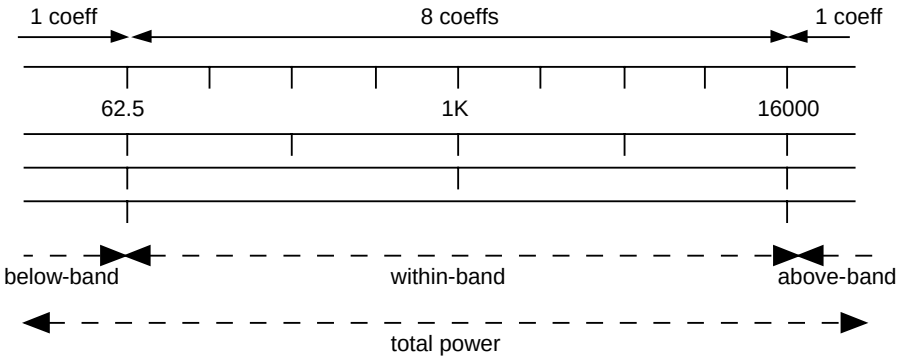

Figure 1 - Illustration of Audio Spectrum Envelope Bands for one-octave bandwidth resolution.

A. De-correlated Spectral Features

Spectrum-based features are often considered canonical for audio applications but it is widely known that direct spectrum features are generally incompatible with classification applications due to their high dimensionality and their inconsistency. Each spectrum slice is an *n*-dimensional vector, with *n* being the number of spectral channels, therefore typical values of a linearly-spaced spectrum are between 64 and 1024 dimensions. Probability classifiers require relatively low-dimensional data representations, preferably fewer than 10 dimensions. A logarithmically-spaced frequency spectrum, such as the one octave bandwidth power spectrum shown in Figure 1, reduces the dimensionality of the representation significantly, but necessarily disregards much information due to the low frequency resolution. What is required is a representation that makes a compromise between dimensionality reduction and information loss.

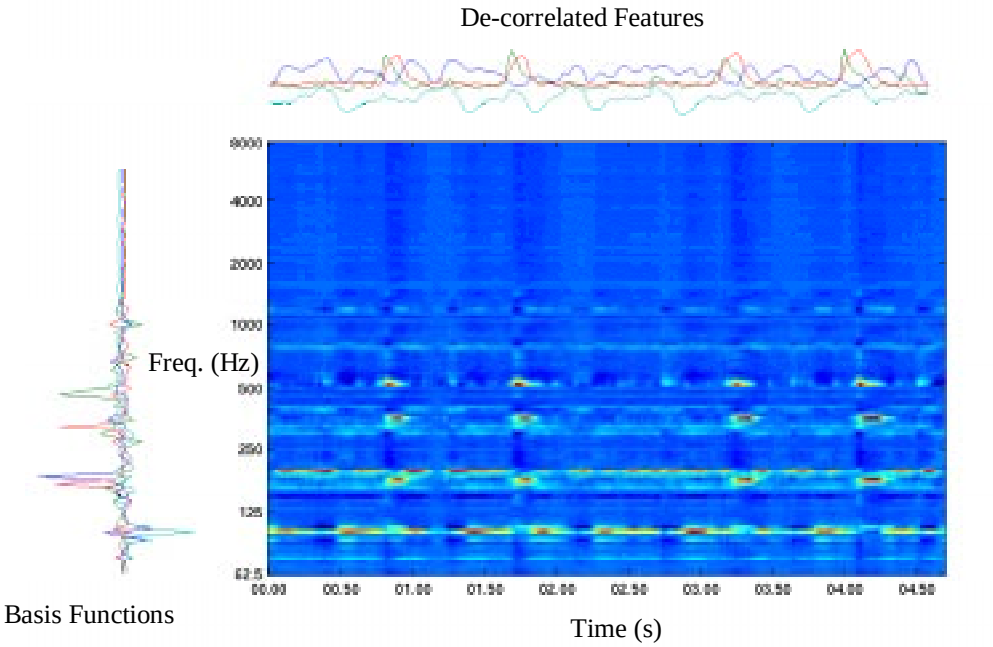

Figure 2. Fitting a spectrogram with data-derived basis functions; in this case the functions were derived from the spectrum using a singular value decomposition (SVD).

A well-known technique for reducing the dimensionality of data whilst retaining maximum information is to use data-derived basis functions, such as computed by *principal component analysis* (PCA), *singular value decomposition* (SVD) or *independent component analysis* (ICA). A spectrogram may be reconstructed using a set of de-correlated frequency basis functions derived using one of these methods. Fewer functions are required to reconstruct a given spectrogram than the total number of frequency channels, hence the possibility for dimensionality reduction. For example, Figure 2 shows a spectrum of five seconds of pop music reconstructed using only four basis functions. The functions on the right of the figure are the frequency basis functions,

those above the figure are the reduced dimension features used for classification. In this case, 70% of the original 32-dimensional data is represented by only the 4-dimensional functions.

The `AudioSpectrumBasis` descriptor contains basis functions that are used to project high-dimensional spectrum descriptions into a low-dimensional representation contained by the `AudioSpectrumProjection` descriptor see DDL Example 2. The reduced bases consist of de-correlated features of the spectrum with the important information described much more efficiently than the direct spectrum representation. This reduced representation is well suited for use with probability model classifiers that typically perform best when the input features consist of fewer than 10 dimensions. These features were found to exhibit superior performance in sound recognition tasks and, thus, we shall describe their extraction and use below.

```
<AudioD xsi:type="AudioSpectrumBasisType" loEdge="62.5" hiEdge="8000"
          resolution="1/4 octave">
    <BasisFunctions>
      <Matrix dim="10 5">
        0.26 -0.05 0.01 -0.70 0.44
        0.34 0.09 0.21 -0.42 -0.05
        0.33 0.15 0.24 -0.05 -0.39
        0.33 0.15 0.24 -0.05 -0.39
        0.27 0.13 0.16 0.24 -0.04
        0.27 0.13 0.16 0.24 -0.04
        0.23 0.13 0.09 0.27 0.24
        0.20 0.13 0.04 0.22 0.40
        0.17 0.11 0.01 0.14 0.37
        0.33 -0.15 0.24 0.05 0.39
     </Matrix>
   </BasisFunctions>
</AudioD>
```

DDL Example 2. Description of five basis functions using AudioSpectrumBasisType. The description definition language is based on XML schema with some extensions specific to MPEG-7. (The floating-point resolution has been truncated for clarity).

B. Spectrum Basis Function Extraction Method

The extraction method for `AudioSpectrumBasis` and `AudioSpectrumProjection` is detailed within the MPEG-7 standard. It is considered that these steps *must* be used in extracting a reduced-dimension description in order to conform to the standard. Within each step there is opportunity for alternate implementations. As such, the following procedure outlines the standardized extraction method for ISA basis functions:

1. *Power spectrum*: instantiate an `AudioSpectrumEnvelope` descriptor using the extraction method defined in `AudioSpectrumEnvelopeType`. The resulting data will be a SeriesOfVectors with *M* frames and *N* frequency bins.

2 *Log-scale norming*: for each spectral vector, **x**, in `AudioSpectrumEnvelope`, convert the power spectrum to a decibel scale:

$$\mathbf{z} = 10\log_{10}(\mathbf{x})$$

and compute the *L2-norm* of the vector elements:

$$r = \sqrt{\sum_{k=1}^{N} z_k^2}$$

the new unit-norm spectral vector is calculated by:

$$\tilde{\mathbf{x}} = \frac{\mathbf{z}}{r}$$

3 *Observation matrix*: place each vector *row-wise* into a matrix. The size of the resulting matrix is *M x N* where *M* is the number of time frames and *N* is the number of frequency bins. The matrix will have the following structure:

$$\tilde{\tilde{\mathbf{X}}} = \begin{bmatrix} \tilde{\tilde{\mathbf{x}}}_1^T \\ \tilde{\tilde{\mathbf{x}}}_2^T \\ \vdots \\ \vdots \\ \tilde{\tilde{\mathbf{x}}}_M^T \end{bmatrix}$$

4 *Basis extraction*: Extract a basis using a singular value decomposition (SVD); commonly implemented as a built-in function in many software packages using the command `[U,S,V] = SVD(X,0)`. Use the *economy* SVD when available since the row-basis functions are not required and this will increase extraction efficiency. The SVD factors the matrix from step 3 in the following way:

$$\tilde{\tilde{\mathbf{X}}} = \mathbf{USV}^T$$

where **X** is factored into the matrix product of three matrices; the row basis **U,** the diagonal singular value matrix **S** and the transposed column basis functions **V.** Reduce the basis by retaining only the first *K* basis functions, i.e. the first *K* columns of **V**:

$$\mathbf{V}_K = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_k \end{bmatrix}$$

K is typically in the range of 3-10 basis functions for feature-based applications. To calculate the proportion of information retained for *K* basis functions use the singular values contained in matrix **S**:

$$I(K) = \frac{\sum_{i=1}^{K} S(i,i)}{\sum_{j=1}^{N} S(j,j)}$$

where *I(K)* is the proportion of information retained for *K* basis functions and *N* is the total number of basis functions which is also equal to the number of spectral bins. The SVD basis functions are stored in the columns of a matrix within the `AudioSpectrumBasisType` descriptor.

6 *Statistically independent basis (Optional)*: after extracting the reduced SVD basis, **V,** a further step consisting of basis rotation to directions of maximal statistical independence is often desirable. This is necessary for displaying independent components of a spectrogram and for any application requiring maximum separation of features.

To find a statistically independent basis using the basis functions obtained in step 4, use one of the well-known, widely published independent component (ICA) algorithms such as INFOMAX, *JADE* or *FastICA;* (Bell and Sejnowski 1995; Cardoso and Laheld 1996; Hyvarinen, 1999).

The ICA basis is the same size as the SVD basis and is stored in the columns of the matrix contained in the `AudioSpectrumBasisType` descriptor. The retained information ratio, *I(K),* is equivalent to the SVD when using the given extraction method.

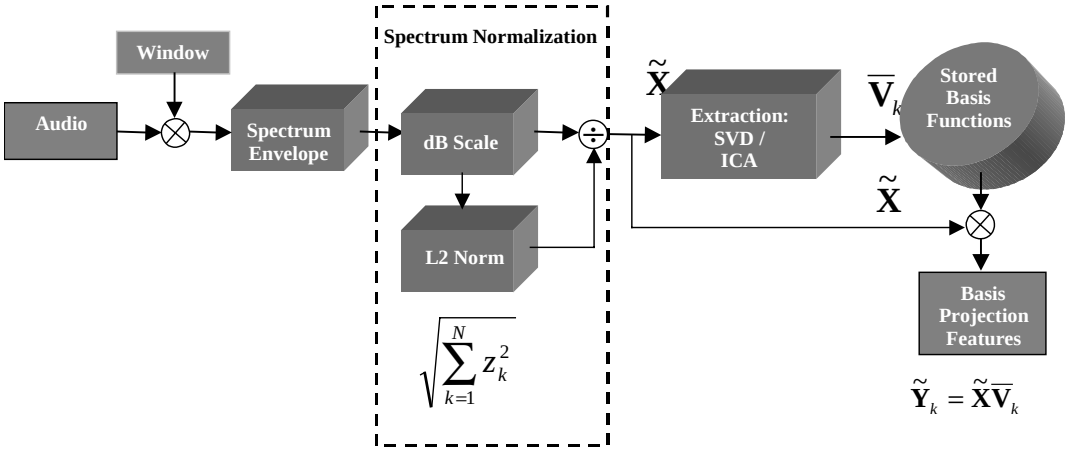

Figure 3 Extraction method for `AudioSpectrumBasis` and `AudioSpectrumProjection.`

C. Basis Projection Features

Figure 3. shows the extraction system diagram for both `AudioSpectrumBasis` and `AudioSpectrumProjection.` The basis projection gives time masking functions that are combined with the spectrum basis functions to reconstruct independent spectrogram components. To perform extraction for `SpectrumBasisProjection` follow steps 1-3 described above for `AudioSpectrumBasis` extraction, this produces a spectrum matrix. The only further requirement is to multiply the spectrum matrix with the basis vectors obtained in step 4 or, optionally, step 5. The method is the same for both SVD and ICA basis functions:

$$\tilde{\mathbf{Y}}_k = \tilde{\mathbf{X}}\overline{\mathbf{V}}_k$$

where **Y** is a matrix consisting of the reduced dimension features after projection of the spectrum against the basis **V**.
For independent spectrogram reconstruction, extract the non-normalized spectrum projection by skipping the normalization step (2) in `AudioSpectrumBasis` extraction. Thus:

$$\mathbf{Y}_k = \mathbf{X}\overline{\mathbf{V}}_k$$

Now, to reconstruct an indpendent spectrogram component use the individual vector pairs, corresponding to the *K*th vector in `AudioSpectrumBasis` and `AudioSpectrumProjection`, and apply the reconstruction equation:

$$\mathbf{X}_k = \mathbf{y}_k \overline{\mathbf{v}}_k^{+}$$

where the + operator indicates the transpose for SVD basis functions (which are orthonormal) or the pseudo-inverse for ICA basis functions (non-orthogonal).

The method outlined above represents a powerful tool that can be used for many purposes. The extracted sources may be subjected to further analysis such as tempo estimation, rhythm analysis or fundamental frequency extraction. For example, we now consider how ISA features may be used for sound recognition and similarity judgements for general audio.

III. GENERALIZED SOUND RECOGNITION

A number of tools exist within the MPEG-7 framework for computing similarity between segments of audio. In this section we describe tools for representing category concepts as well as tools for computing similarity in a general manner. The method involves training statistical models to learn to recognize the classes of sound defined in a taxonomy.

A. Taxonomies

A taxonomy consists of a number of sound categories organized into a hierarchical tree. For example, voice, instruments, environmental sounds, animals, etc. Each of these classes can be broken down further into more detailed descriptions such as: female laughter, rain, explosions, birds, dogs, etc.

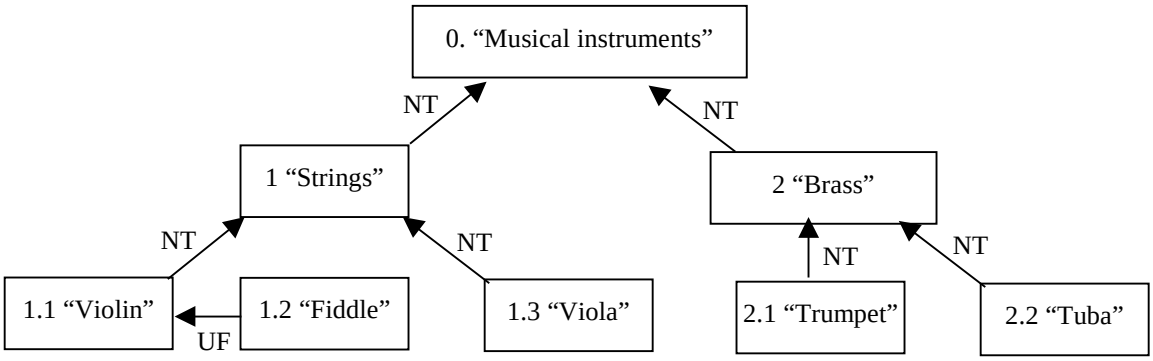

Figure 4. A controlled-term taxonomy of part of the *Musical Instruments* hierarchy

Figure 4 shows musical instrument controlled terms that are organized into a taxonomy with “Strings” and “Brass”. Each term has at least one relation link to another term. By default, a contained term is considered a narrower term (NT) than the containing term. However, in this example, “Fiddle” is defined as being a nearly synonymous with, but less preferable than, “Violin”. To capture such structure, the following relations are available as part of the `ControlledTerm` description scheme:

- *BT* – **Broader term**. The related term is more general in meaning than the containing term.
- *NT* – **Narrower term**. The related term is more specific in meaning than the containing term.
- *US* – **Use** The related term is (nearly) synonymous with the current term but the related term is preferred to the current term.
- *UF* – **Use for**. Use of the current term is preferred to the use of the (nearly) synonymous related term.
- *RT* – **Related Term**. Related term is not a synonym, quasi-synonym, broader or narrower term, but is associated with the containing term.

The purpose of the taxonomy is to provide semantic relationships between categories. As the taxonomy gets larger and more fully connected the utility of the category relationships increases. Figure 5 shows the taxonomy in Figure 4 combined into a larger classification scheme including animal sounds, musical instruments, Foley sounds (sound effects for film and television), and impact sounds. By descending the hierarchical tree we find that there are 17 leaf nodes in the taxonomy. By inference, a sound segment that is classified in one of the leaf nodes inherits the category label of its parent node in the taxonomy. For example, a sound classified as a “Dog Bark” also inherits the label “Animals”. We shall adhere to this taxonomy for illustrative purposes only; MPEG-7 allows full flexibility in defining taxonomies using controlled terms and can be used to define much larger taxonomies than the given example.

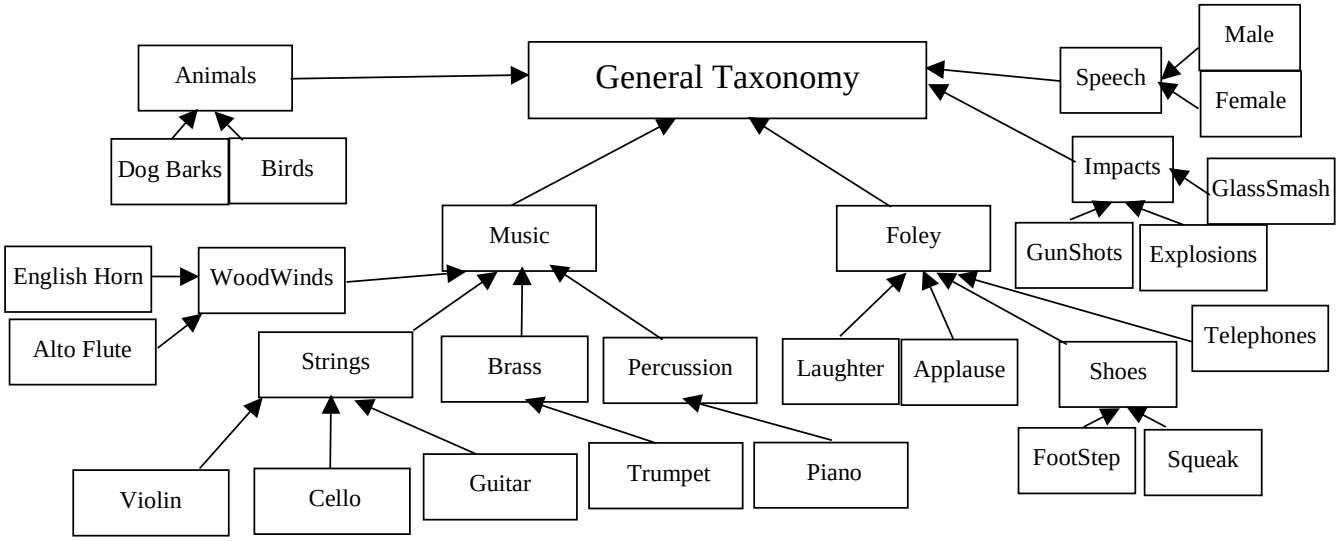

Figure 5. A hierarchical taxonomy including both musical and non-musical sources.

B. Probability Model Classifiers

A number of probability model description schemes are defined in MPEG-7. The motivation for standardization is driven by the cost of designing and training a robust classifier; which can be very computationally intensive for large-scale applications. Statistical models for classification of content may be shared via a standard interface to the internal probability models. With these standardized schemes in hand, it is possible to share pre-trained probability models between applications even if the specific extraction methods vary, thus allowing widespread re-use of models. The following sections outline the use of probability model description schemes for sound recognition.

1) Finite State Models

Sound phenomena are dynamic. The spectral features vary in time and it is this variation that gives a sound its characteristic fingerprint for recognition. MPEG-7 sound-recognition models partition a sound class into a finite number of states based on the spectral features; individual sounds are described by their trajectories through this state space. Each state is modeled by a continuous probability distribution such as a Gaussian.

The dynamic behaviour of a sound class through the state space is modeled by a *k* x *k* transition matrix that describes the probability of transition to each of the *k* states in a model given a current state. For a transition matrix, **T**, the *i*th row and *j*th column entry is the probability of transitioning to state *j* at time *t* given state *i* at time *t-1*.

An initial state distribution, which is a *k* x *1* vector of probabilities, is also required for a finite-state model. The *k*th element in the vector is the probability of being in state *k* in the first observation frame.

2) Multi-dimensional Gaussian Distributions

The multi-dimensional Gaussian distribution is used for modeling the states. Gaussian distributions are parameterized by a *1 x n* vector of means, **m**, and an *n x n* covariance matrix, **K**, where *n* is the number of features (columns) in the sound observation vectors. The expression for computation of probabilities for a random column vector, **x**, given the Gaussian parameters is:

$$f_{\mathbf{x}}(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\mathbf{K}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mathbf{m})^T \mathbf{K}^{-1}(\mathbf{x}-\mathbf{m})\right].$$

3) Continuous Hidden Markov Models

A continuous hidden Markov model is a finite state model with Gaussian distributions approximating each state's probability distribution. The states are *hidden* since we are not given the states along with the data. Rather, we must use the observable data

to infer the hidden states. The states are clusters in the feature space of the sound data; namely, the `SpectrumBasisProjection` audio descriptor discussed earlier. Each row of the projected feature matrix, defined above, is a point in an n-dimensional vector space. The cloud of points is divided into multiple states (Gaussian clusters) using maximum *a posteriori* (MAP) estimation. The MAP estimator has the property of minimizing the entropy, or degree of uncertainty, of the model whilst maximizing the number of bits of evidence (information) that supports each model parameter, (Brand,1998; Brand 1999) Figure 6 shows a representation of 4 Gaussian-distributed states (vector point clouds) in two dimensions.

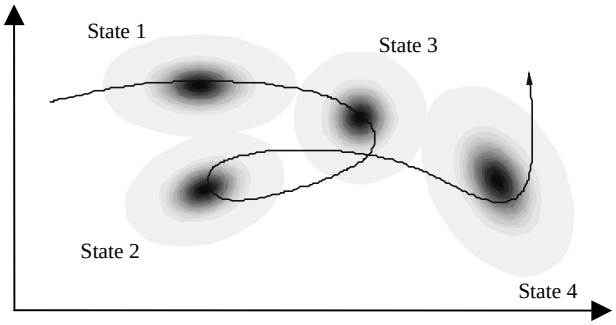

Figure 6. Four estimated Gaussian states depicted in a two-dimensional vector space. Darker regions have higher probabilities. Sounds are represented as trajectories in such a vector space, the states are chosen to maximize the probability of the model given the observable evidence; i.e. the training data. The line shows a possible trajectory of a sound vector through the space.

4) MPEG-7 representation of hidden Markov models

DDL example 3 illustrates the use of probability model description schemes for representing a continuous hidden Markov model with Gaussian states; in this example floating-point numbers have been rounded to 2 decimal places for display purposes only.

```
<ProbabilityModel xsi:type="ContinuousMarkovModelType" numberStates="7">
<Initial dim="7">
0.04  0.34  0.12  0.04  0.34  0.12  0.00 </Initial>
<Transitions dim="7 7">
0.91  0.02  0.00  0.00  0.05  0.01  0.01
0.01  0.99  0.00  0.00  0.00  0.00  0.00
0.01  0.00  0.92  0.01  0.01  0.06  0.00
0.00  0.00  0.00  0.99  0.01  0.00  0.00
0.02  0.00  0.00  0.00  0.97  0.00  0.00
0.00  0.00  0.01  0.00  0.00  0.98  0.01
0.02  0.00  0.00  0.00  0.00  0.02  0.96
</Transitions>
<State><Label>1</Label></State>
<!—State 1 Observation Distribution -- >
<ObservationDistribution xsi:type="GaussianDistributionType">
<Mean dim="6">
5.11 -9.28 -0.69 -0.79  0.38  0.47
</Mean>
<Covariance dim="6 6">
1.40 -0.12 -1.53 -0.72  0.09 -1.26
-0.12  0.19  0.02 -0.21  0.23  0.17
-1.53  0.02  2.44  1.41 -0.30  1.69
-0.72 -0.21  1.41  2.27 -0.15  1.05
0.09  0.23 -0.30 -0.15  0.80  0.29
-1.26  0.17  1.69  1.05  0.29  2.24
</Covariance>
<State><Label>2</Label></State>
<!—Remaining states use same structures-- >
<\PobabilityModel>
```

DDL Example 3. Instantiation of a Probability Model in the MPEG-7 DDL language. The model parameters were extracted using a maximum *a posteriori* estimator. The description scheme represents the initial state distribution, transition matrix, state labels, and individual Gaussian means and covariance matrices for the states.

IV. Sound Classification, Similarity and Example Search Applications

A. Classification Application

We trained 19 HMMs, using MAP estimation, on a large database (2000+ sounds) divided into 19 sound classes as described by the leaf nodes in the general sound taxonomy shown in Figure 5 above. The database was split into separate training and testing data sets. That is, 70% of the sounds were used for training the HMM models and 30% were used to test the recognition performance of the models on novel data. Each sound in the test set was presented to all 19 models in parallel, the HMM with the maximum likelihood score, using a method called Viterbi decoding, was selected as the representative class for the test sound; see Figure 7.

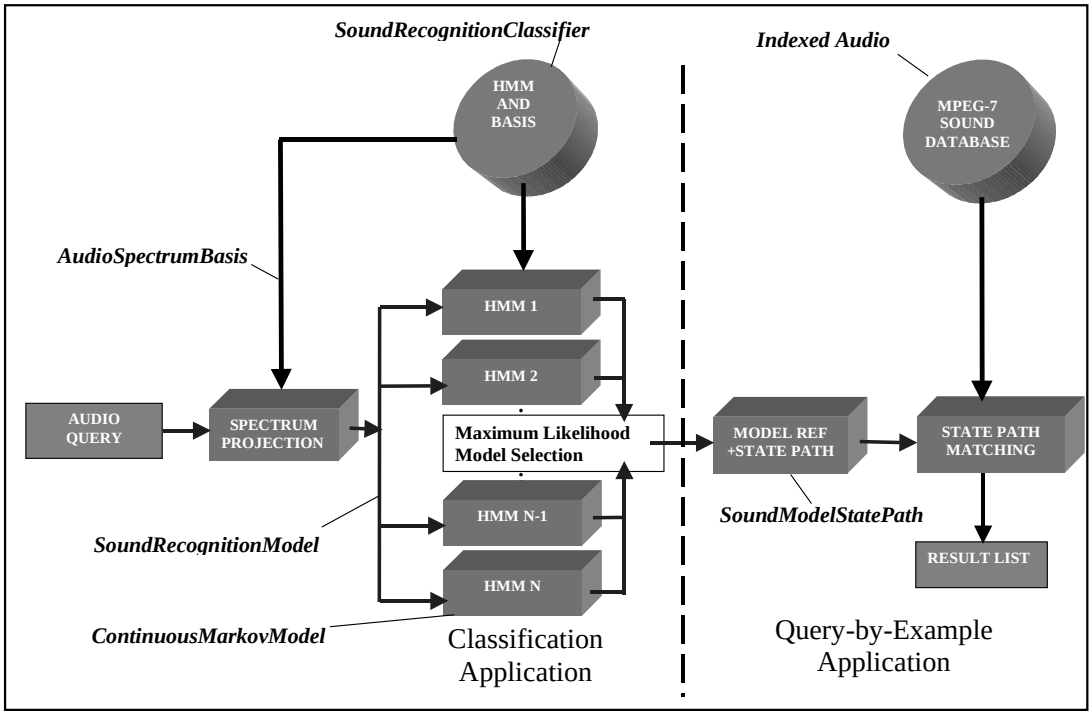

Figure 7: Sound Classification and Similarity System Using Parallel HMMs

The results of classification performance on testing data are shown in Table 1. The results indicate very good recognizer performance across a broad range of sound classes. Of particular note is the ability of the classifiers to discriminate between speech sounds and non-speech sounds including distinguishing between male and female speakers. The between class discrimination indicates a high degree of category resolution for the system.

Table 1. Performance of 19 classifiers trained on 70% and cross-validated on 30% of a large sound database. The mean recognition rate indicates high recognizer performance across all the models..

Model Name	% Correct Classification
[1] AltoFlute	100.00
[2] Birds	80.00
[3] Pianos (Bosendorfer)	100.00
[4] Cellos (Pizz and Bowed)	100.00
[5] Applause	83.30
[6] Dog Barks	100.00
[7] English Horn	100.00
[8] Explosions	100.00
[9] Footsteps	90.90
[10] Glass Smashes	92.30

[11] Guitars	100.00
[12] Gun shots	92.30
[13] Shoes (squeaks)	100.00
[14] Laughter	94.40
[15] Telephones	66.70
[16] Trumpets	80.00
[17] Violins	83.30
[18] Male Speech	100.00
[19] Female Speech	97.00
Mean Recognition Rate	**92.646**

In addition to testing on the general sound taxonomy, we also conducted an experiment in music genre classification using the feature extreaction and training methodology outlined above. We collected several hours of material from compact discs and MPEG-1 (Layer III) audio compressed files representing eight different musical genres. The data was split into 70%/30% training/testing sets and the hidden Markov model classification system was trained as detailed above. Each sound file was split into separate 30 second segments for training, thus the models were tuned to capture both local beat structure as well as phrase structures inherent within the data. Table 2 shows the results of classification of musical genres on novel musical sound file data. These results indicate the generalization of the classification system to sounds that are composed of mixed sources with a high degree of internal structure.

Table 2. Performance of 8 classifiers using a 70%/30% training/testing split for music genre classification.

Model Name	**% Correct Classification (novel data)**
[1] Bluegrass	96.8
[2] Reggae	92.5
[3] Rap	100.0
[4] Folk	92.3
[5] Blues	98.7
[6] Country	88.9
[7] Gospel	95.7
[8] NewAge	98.3
Mean Recognition rate	**95.4%**

B. Generalized Sound Similarity

In addition to classification, it is often useful to obtain a measure of how *close* two given sounds are in some perceptual sense. It is possible to leverage the internal, hidden, variables generated by an HMM in order to compare the evolution of two sounds through the model's state space. For each input query sound to a HMM, the output is a series of states through which sound passed. Each sampled state is given a *likelihood* that is used to cumulatively compute the probability that the sound actually belongs to the given model. The `SoundModelStatePath` descriptor contains the dynamic state path of a sound through a HMM model. Sounds are indexed by segmentation into model states or by sampling of the state path at regular intervals. Figure 8 shows a spectrogram of a dog bark sound with the state path through the "DogBark" HMM shown below.

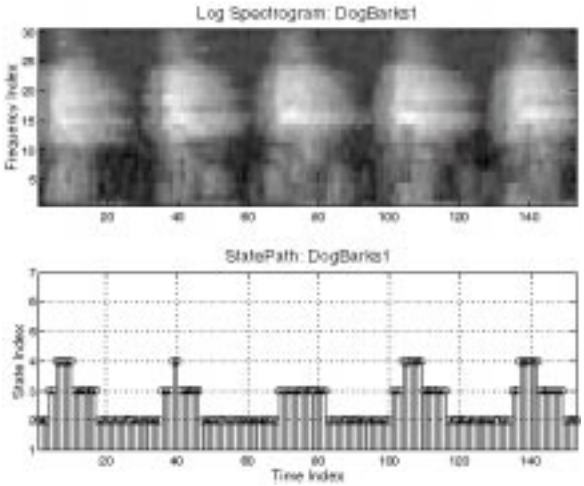

Figure 8. Dog bark spectrogram and the state path through the dog bark continuous hidden Markov model

The state path is an important method of description since it describes the evolution of a sound with respect to physical states. The state path shown in the figure indicates physical states for the dog bark; there are clearly delimited onset, sustain and termination/silent states. This is true of most sound classes; the individual states within the class can be inspected via the state path representation and a useful semantic interpretation can often be inferred.

There are many possible methods for computing similarity between state paths; dynamic time warping and state histogram sum-of-square errors are two such methods. Dynamic time warping (DTW) uses linear programming to give a distance between two functions in terms of the cost of warping one onto the other. We may apply DTW to the state paths of two sounds in order to estimate the similarity of their temporal evolutions. However, there are many cases where the temporal evolution is not as important as the relative balance of occupied states between sounds. This is true, for example, with sound textures such as rain, clapping or crowd babble. For these cases it is preferable to use a temporally agnostic similarity metric such as the sum-of-square errors between state occupancy histograms. These similarity methods may be applied to a wide variety of sound classes and thus constitute a generalized sound similarity framework.

C. Query-by-Example Application

The system shown in the right-hand side of Figure 7 implements a query-by-example application. The audio feature extraction process is applied to a target query sound, namely spectrogram projection against a stored set of basis functions for each model in the classifier. The resulting dimension-reduced features are passed to a Viterbi decoder for the given classifier and the HMM with the maximum-likelihood score for the given features is selected. The model reference and state path are recorded and the results are matched by comparing the state path to the state paths of all the sounds for the given class in a pre-computed MPEG-7 index database.

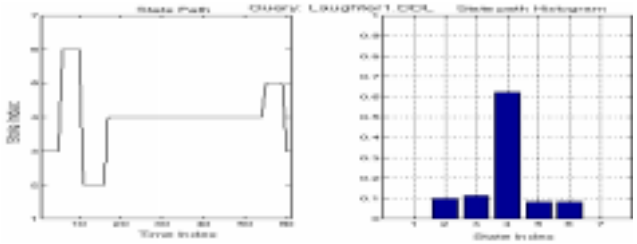

Figure 9. query sound represented by a state-path histogram for the Laughter HMM.

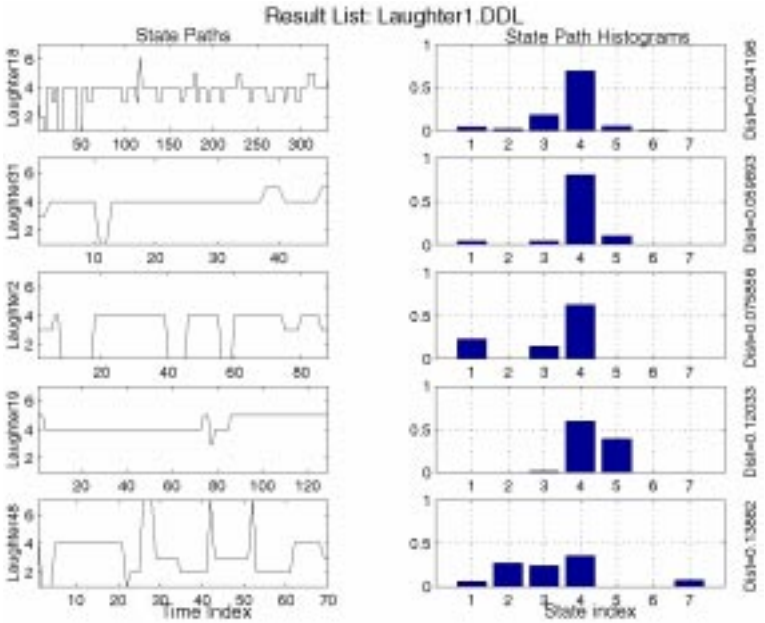

Figure 10. 5-best matches for the query sound. The distances between the target sound and the result sounds are given on the right-hand side of the figure. These distances were computed using the sum of square errors between the state-path histograms.

Figure 9 shows a query sound (Laughter) and Figure 10 shows the resulting closest matches using the difference in state-path occupancy histograms. The state paths and the histograms are also shown in the figures as well as the resulting distance estimates for each of the returned matches.

D. Non-Categorical Similarity Ratings

Using such similarity measures it is possible to automatically organize sonic materials for a composition. The examples given above organize similarity rankings according to a taxonomy of categories. However, if a non-categorical interpretation of similarity is required one may simply train a single HMM, with many states, using a wide variety of sounds. Similarity may then proceed without category constraints by comparing state-path histograms in the large generalized HMM state space.

V. CONCLUSIONS

In this paper we have outlined some of the tools that are available within the MPEG-7 standard for managing complex sound content. In the first part of the paper we presented independent subspace analysis as a method for performing analysis and re-synthesis of individual sources in a mixed audio file. We also showed that ISA may be used to obtain statistically salient features that may be applied with great generality to sound recognition and sound similarity tasks.

One of the major design criteria for the tools was the ability to analyze and represent a wide range of acoustic sources including textures and mixtures of sound. The tools presented herein exhibited good performance on musical sounds as well as traditionally non-musical sources such as vocal utterances, animal sounds, environmental sounds and sound effects. Amongst the applications presented were robust sound recognition using trained probability model classifiers and sound similarity matching using internal probability model state variables.

In conclusion, the description schemes and extractor methodologies outlined in this paper provide a consistent framework for analyzing, indexing and querying sounds from a wide range of different classes. These tools have been made widely available as a component of the reference software implementation of the MPEG-7 standard. It is hoped that the ability to manipulate sound in novel ways and the ability to search for “sounds like” candidates in a large database of sounds will become important new tools for sound-designers, composers and many other users of new music technology.

References

Bell, A. J. and Sejnowski, T.J. 1995. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation,* 7:1129-1159.

Boreczky, J.S. and Wilcox, L.D. 1998. A hidden Markov model framework for video segmentation using audio and image features, in *Proceedings of ICASSP'98,* pp.3741-3744, Seattle, WA.

Brand, M. 1998. Structure discovery in conditional probability models via an entropic prior and parameter extinction. *Neural Computation.*

Brand, M. 1999. Pattern discovery via entropy minimization. In *Proceedings, Uncertainty'99.* Society of Artificial intelligence and Statistics #7. Morgan Kaufmann.

Cardoso, J.F. and Laheld, B.H. 1996. Equivariant adaptive source separation. *IEEE Trans. On Signal Processing,* 4:112-114.

Casey, M.A., and Westner, A. 2000. Separation of mixed audio sources by independent subspace analysis. *Proceedings of the International Computer Music Conference,* ICMA, Berlin.

Casey, M. 2001a. "MPEG-7 Sound Recognition Tools", *IEEE Transaction on Circuits and Systems Video Technology*, special issue on MPEG-7, IEEE.

Casey, M. 2001b. "Sound Classification and Similarity Tools", in B.S. Manjunath, P. Salembier and T. Sikora, (Eds), *Introduction to MPEG-7: Multimedia Content Description Language*, J. Wiley, 2001

Casey, M. 2001c. "Reduced-Rank Spectra and Entropic Priors as Consistent and Reliable Cues for General Sound Recognition", *Proceedings of the Workshop on Consistent & Reliable Acoustic Cues for Sound Analysis*, September, 2001.

Hyvarinen, A. 1999. Fast and robust fixed-point algorithms for independent component analysis. *IEEE Trans. On Neural Networks,* 10(3):626-634.

International Standards Organization (ISO). 2001, *Information Technology – Multimedia Content Description Interface – Part 4: Audio*. ISO-IEC 15938-4 (E).

Martin, K. D. and Kim, Y. E. 1998. Musical instrument identification: a pattern-recognition approach. Presented at the 136th Meeting of the Acoustical Society of America, Norfolk, VA.

Wold, E., Blum, T., Keislar, D., and Wheaton, J. 1996. Content-based classification, search and retrieval of audio. *IEEE Multimedia,* pp.27-36, Fall.

Zhang, T. and Kuo, C. 1998. Content-based classification and retrieval of audio. SPIE 43rd Annual Meeting, *Conference on Advanced Signal Processing Algorithms, Architectures and Implementations VIII,* San Diego, CA.