

# Scalable, Content-Based Audio Identification by Multiple Independent Psychoacoustic Matching\*

GEOFF R. SCHMIDT\*\*

*Intellivid Corporation, Cambridge, MA 02138-1171, USA*

AND

MATTHEW K. BELMONTE\*\*

*Departments of Psychiatry and Experimental Psychology, University of Cambridge, Cambridge CB2 2AH, UK*

A software system for content-based identification of audio recordings is presented. The system transforms its input using a perceptual model of the human auditory system, making its output robust to lossy compression and to other distortions. In order to make use of both the instantaneous pattern of a recording's perceptual features and the information contained in the evolution of these features over time, the system first matches fragments of the input against a database of fragments of known recordings. In a subsequent step, these matches at the fragment level are assembled in order to identify a single recording that matches consistently over time. In a small-scale test the system has matched all queries successfully against a database of 100 000 commercially released recordings.

## 0 INTRODUCTION

The field of informatics is in the midst of a transformation from purely textual systems, in which indexing is driven by tried-and-true methods of string matching, to multimedia systems, in which measures of similarity are more dimensionally complex and computationally intensive. As the capacity for information storage has outpaced developments in algorithms, indexing of pictures and sounds has been left to rely not on the actual records being indexed, but rather on file names or other textual labels attached to these records. Anyone who has made use of image search engines or peer-to-peer file sharing systems knows that these labels, or metadata, inevitably fail to capture essential information. The top matches returned by a search may turn out to be altered editions related to the target item (for example, a live concert recording, a remix, or a cover), or even the wrong item altogether.

To escape this dependence on fallible metadata, search algorithms are needed that are keyed not on attached text

but on the actual data being indexed. The complexity inherent in such algorithms is a product of the fact that perceptual similarities appear not so much in a media file's raw data as in its many derived properties. Similarities apparent to the human senses are seldom evident in comparisons of the data's raw bytes. Images whose actual pixel values are utterly different from each other may nonetheless look alike to the human visual system, and sounds whose time-series data are uncorrelated may nonetheless sound alike to the human auditory system. Samples may vary along multiple perceptual axes, making the search space high-dimensional and therefore making nearest-neighbor searches exponentially more difficult. A simple example of this dimensionality problem is a time-frequency representation of a sound, in which the number of dimensions is the product of the number of time steps and the number of frequency bands. In order for the problem of comparing media files to be rendered tractable, the dimensionality of the inputs must be reduced. The process of deterministically computing a relatively low-dimensional, unique identifier for a media file is known as the fingerprinting problem. Fingerprinting in the audio domain, in particular, has received a great deal of attention due to its immediate applications in archive indexing and searching, automated broadcast monitoring, and digital rights management.

---

\*Manuscript received 2003 August 12; revised 2003 December 30.

\*\*Formerly with Tuneprint, Inc., Cambridge, MA 02139-4056, USA.

The audio fingerprinting schemes published to date extract information by applying frequency-domain analyses within narrow temporal windows. Some of these methods use a straightforward time–frequency representation as input to further processing steps [1]–[6] whereas others use the Fourier spectrum to compute derived measures such as modulation frequency [7], measures of spectral shape and tonality [8], [9], MPEG-7 feature sets [10], or hash bits based on frequency-specific temporal changes [11], [12]. One published method extracts individual notes and then applies conventional string-matching algorithms to sequences of these notes [13], though of course such a method does not allow for complexities such as superpositions of various instruments or vocals. All of these systems either depend on supervised learning algorithms applied to fairly raw time–frequency representations, or decide a priori what specific spectral features or measures will be relevant. In the former case, perceptually relevant information is submerged in a large corpus of data, making statistical learning algorithms liable to discriminate on the basis of perceptually irrelevant features. In the latter case, feature extraction discards a great deal of useful information along with the irrelevant details. Either way, classification is degraded.

In addition to this loss of perceptual information within time points, many existing techniques do not make full use of information on changes in a recording's perceptual qualities across time points. A single recording may evolve temporally through many different styles, tempos, and timbres. Previous audio fingerprinting methods have suffered from a needlessly exclusive view of time and frequency representations, collapsing localized frequency-domain features across long intervals of time. This strategy accomplishes a great deal of data reduction, at the expense of blurring out short-lived properties that could be useful for identification. The Muscle Fish system [8], for example, computes feature vectors for a large set of closely spaced time frames within a recording, but then retains only the mean, variance, and autocorrelation of these feature vectors over the entire recording. This strategy works well for brief samples such as sound effects [8] but is unlikely to scale well to temporally extended recordings. A system described recently by Sukittanon and Atlas [7] adopts a similar approach, computing subband modulation frequencies in each frame and then preserving only the centroids of these features across frames. A system described by Burges et al. [6] operates only on brief clips, basing its classifications solely on the one frame that differs least from a frame in the database, without using information from surrounding frames. Other systems [2], [4] pool feature vectors across the entire recording into a histogram of vector-quantization bins, destroying temporal sequence information as the data from the individual vectors are summed into the histogram. Another system based on vector quantization [9] sums the error between feature vectors and vector quantization codebook entries, and then classifies the recording as a whole on the basis of the codebook entry whose summed error is least. Here again, information on the recording's time course is lost in the summation.

One way around the problem of temporal blurring in pooled data is to sum not the match scores themselves but rather the outputs of some nonlinear “squashing function” applied to the match scores. Good matches of short-lived features will thus be given a disproportionate effect on the match quality of the recording as a whole. A system designed by Papaodysseus et al. [5] adopts this approach using a simple step function as a squashing function, that is, matches at each time point are thresholded, and the summed match score is incremented only if the match at the current time point is above threshold. With this approach also, though, useful data are lost, since information on partial, subthreshold matches within individual time points has no effect on matching the recording as a whole, even when such partial matches occur at a large number of time points.

A difficulty in applying information from partial matches is the time complexity of searching for these matches. Exact matches, on the other hand, are much more efficiently identified. Quantization of the signal's features will produce some number of exact matches with identically quantized records in the database. The database recordings that generated these exact matches can then be searched for approximate matches. Constraining the search space in this way renders the approximate search problem tractable. This strategy has been applied in a hashing system based on temporal differences in subband amplitude differences (that is, a double differentiation in frequency and time) [11], [12]. Such a search method preserves information across time points, though it remains an open question whether this method of time–frequency differentiation can be improved on for extracting perceptually relevant information within time points.

Losses of useful data within and across time points are likely a major reason why fingerprinting schemes in general have suffered from error rates that would be unacceptable in any large-scale system. Though more advanced systems (such as [12]) promise improved results, most of the methods cited feature rates of successful matching in the range of 90 to 99%. When matching against a database of hundreds of thousands or even millions of recordings, even 99% is unacceptable. Worsening the outlook for scalability is the fact that many of these error rates arise from tests against small databases on the order of hundreds [8], [1]–[4] or thousands [13], [9], [7] of recordings, or within restricted musical genres [1], [3]. In order to improve performance, strategies must be developed to extract selectively the perceptually relevant information within time points, and to preserve this information across time points. For both of these goals one can look to human neurobiology as a model.

The goal of preservation of feature-specific information across time points has been addressed in a biologically motivated neural network model proposed by Hopfield and Brody [14]. This model recognizes audio signals via a massively parallel network of independent feature detectors whose outputs decay with various time constants. The recognition signal consists of a selectively weighted sum of these many time-dependent, feature-dependent measures. Such a network of multiple independent feature de-

tectors is the parallel-processing equivalent of a serial strategy involving multiple independent matching of features at a series of time points, followed by an evaluation of the match results for temporal consistency. This strategy makes the recognition problem tractable by separating the problem of matching within time points from the problem of matching across time points. Matching within time points produces in general some scalar measure of goodness of fit. Matching across time points then amounts to an instance of the well-known problem of weighted linear regression, where the time indices of an unknown input recording are regressed against the time indices of a reference recording using these within-time-points measures as weights, and a regression line of unit (or near-unit) slope indicates a match. This method of weighted linear regression has been applied to audio data by Schmidt [15], [16] and independently by Wang [17].

The choice of a method for extracting perceptual information within time points depends on the context for which the audio fingerprinting system has been designed. While some systems [4] only classify the input as to musical genre or class, others rank better and worse matches within a class [8], or identify single best matches [13], [2], [9], [5], [7]. Among the systems designed for individual identification, variously inclusive or expansive criteria can be established. The most narrow and least useful sense of identity is the simple equality of two signals. In this case, byte-for-byte comparison suffices and no fingerprinting is needed. At the opposite extreme, some applications may need to establish broad identity, retrieving sounds that contain similar sequences or thematic elements but different instrumentals or voices (such as remixes or covers). Perhaps most useful for applications of digital rights management, though, is an audio indexing system that identifies an input as one record within a database of releases, after that input has perhaps been distorted by slight variations in playback speed, by lossy compression, or by transmission through a bandwidth-limited system. Such an indexing system would identify recordings that sound the same to a human listener, and would differentiate recordings that contain obvious differences.

This criterion of equivalence to a human listener leads to a natural approach to enhancing perceptually relevant information within time points: if the software is made to model human auditory processing, then features not important in human auditory discrimination will be lost, and features that contribute to discrimination will be retained. As a result, the criterion of equivalence to a human listener can be attained without any assumptions as to what particular features are relevant. A classifier based on human auditory modeling is in a way an elaboration of time-frequency methods that partition the audible spectrum into frequency bands akin to the critical bands of the human cochlea [2], [4], [5]. One such method has achieved 99.8% recognition by applying dimensionality reduction and supervised learning after accounting for frequency-specific human auditory perceptual thresholds [6], though more complex psychoacoustic phenomena such as spreading and compression are not accounted for.

In order to preserve information across time points, we adopt the philosophy that many approximate tests of matching are better than a single, make-or-break test. Other systems have constructed databases whose elements are whole recordings, and this strategy leads inevitably to the problem of pooling data across time points. In contrast, the system discussed in this study, known as Tuneprint, matches each short fragment of a recording against a large database consisting of every individual fragment of every recording loaded. Fast vector quantization within such a large database is inevitably fallible, and thus many fragments within an input recording will produce incorrect matches. A significant portion of matches, however, will be correct, and it is this redundancy across time points that is the key to the Tuneprint algorithm: if significantly many fragments from the input produce matches that turn out to be the temporally corresponding fragments from a single recording, the input can be identified as this recording. Using a sophisticated psychoacoustic model as the front end to this fragment-based analysis, Tuneprint has achieved a high success rate on a database of 100 000 commercial music releases.

## 1 THE ALGORITHM

### 1.1 General Considerations

The identification algorithm can be conceptualized as the combination of a perceptual model, which eliminates features that are not significant to a human listener, a fragment function, which matches each of the input's individual temporal fragments against fragments in the database, and an assembly function, which puts together results from the fragment function over time and evaluates their consistency. In general, a fragment function takes an input recording and a temporal offset within that recording, and produces a set of triples, each consisting of a recording in the database, a temporal offset within that recording, and a distance measure or some other quantification of match quality. An assembly function takes a series of the fragment function's outputs over time, and produces a set of outputs each of which associates a recording in the database with a confidence level with which the input matches that recording.

The goal in defining the fragment function is not so much to maximize accuracy as to maximize the information garnered per unit of computational resources spent. Depending on how the fragment function is implemented, the limiting resource may be input-output bandwidth, memory bandwidth, network bandwidth, or CPU time. It is expected that a fragment function will be noisy, even returning no information at all for some particularly difficult cases. The assembly function is able to extract a signal from this noisy output by exploiting the constraint that any genuine match must be consistent over an interval of time.

Since matching is evaluated independently for each fragment, it is possible for a match to occur between any subinterval of a recording submitted for identification and any matching subinterval of a recording in the database. This ability to construct independent matches on subintervals makes the Tuneprint system robust to truncation of

audio, to the addition of silent intervals, and to momentary glitches such as sometimes occur during radio or streaming transmission. Useful results can be obtained even from less than a second of input.

## 1.2 Psychoacoustic Modeling

As a front end to Tuneprint's fragment function, audio recordings are transformed by a psychoacoustically based model of human hearing. Input is sampled at 44.1 kHz, either from raw CD content or by playback from a compressed format. In the case of a stereo recording, left and right channels are mixed to mono. Intervals of 185.715 ms (8190 samples) are normalized for playback volume by subtracting the mean and then scaling to the interval's maximum excursion or to one-sixteenth of the playback medium's dynamic range, whichever is greater. (This dynamic range criterion prevents silent intervals from becoming high-amplitude noise.) The interval is multiplied by a Hann window, padded with a single zero on each end, and then Fourier-transformed. A power spectrum with a resolution of 5.38 Hz is extracted from the Fourier transform, over the frequency range from 253 to 12 500 Hz. Frequencies outside this range are discarded so that matches cannot depend on them. We have found this strategy effective at improving the identification of band-limited transmissions while having no negative effect on the identification of full-bandwidth recordings. The power spectrum is transformed from a hertz scale to a Bark scale by linear interpolation using the Bark frequency values given in [18]. This transformation yields a power spectrum extending from 2.53 Bark to 23.17 Bark in 128 discrete steps, energy from multiple frequency bins at the high end of the spectrum being summed into single Bark bins.

Transformation to the Bark scale sets the stage for the computation of frequency spreading. In the human ear, mechanical properties of the basilar membrane and coarse coding within the cochlear nucleus cause a single-frequency input to excite neurons encoding a range of frequencies, with a central peak of excitation occurring at the input frequency. This spreading of neural excitation across frequencies underlies the psychoacoustic phenomenon of masking, in which a low-amplitude tone, at a frequency near that of a higher amplitude tone or a band of noise, cannot be resolved [18]. Most applications based on human perceptual modeling (for example, MPEG layer 3 [19]) compute an intensity threshold below which a masked sound will not be heard. Since thresholds differ depending on whether the masker is a pure tone or noise, this thresholding method has the disadvantage of relying on rather arbitrary measures of spectral flatness as indices of tonality.

An alternative to modeling the masking threshold is to model the frequency spreading itself. With this approach, the masked threshold is not explicitly computed. Instead, it arises from decreases in discriminability due to spreading of the spectrum. The computational distinction between noise masking and tone masking also is obviated [20]: the same model is used in both cases, and the difference arises in the model's behavior in response to masking inputs whose levels are fairly uniform across frequencies (noise

maskers) versus those whose levels are frequency-specific (tone maskers). As specified in [20], we spread each Bark band with a left-sided rolloff of 31 dB/Bark and a right-sided rolloff of 22 dB/Bark + (230 Hz/Bark)/ $\nu$ , where  $\nu$  is the frequency (in hertz) of the masker. We apply an intensity compression factor  $\alpha$  of 0.8, making the compressed sum of excitations greater than the linear sum. Algorithmic details on the application of this compression factor and the computation of spreading are presented in [20]. In addition to accounting for masking, this spreading of spectral peaks makes the identification robust to discretization errors that may arise from small variations in playback speed.

Following spreading and conversion to an intensity (dB) scale, the minimum audible field (MAF) [21], expressed in dB as a function of Bark frequency, is subtracted from the signal. Bark frequency bands in which this subtraction yields a negative result are zeroed, whereas nonnegative results are transformed according to a perceptual loudness measure based on that defined in [22], depending on Bark frequency  $z$  and frequency-specific intensity  $I_z$ :

$$0.7 + 0.4/\text{Bark} \cdot (2.5 \text{ Bark} - \max[2.5 \text{ Bark}, \min(3 \text{ Bark}, z)]) \cdot (I_z - 100 \text{ dB}) + 100 \text{ dB} + 8.5 \text{ dB} \cdot \left[ 1 - \frac{1}{1 + e^{-0.09/\text{dB} \cdot (I_z - 60 \text{ dB})}} \right].$$

This step completes the psychoacoustic transformation, an example of which is shown in Fig. 1. The resulting output changes fairly slowly across samples (Fig. 2), making Tuneprint robust to temporal frame shifts.

As a postprocessing step, the spectrum is high-pass filtered by subtracting from each point the linear trend in the 6-Bark interval centered on that point. (In the case of points that lie within 3 Bark of the spectrum's upper or lower edge, the detrending window is narrowed accordingly.) Although this last filtering step may seem to diverge from the goal of modeling human perception, we find it useful in practice since it removes band-limited intensity offsets that arise from equalization (see example in Fig. 3). Such equalization can be produced deliberately, but is more often an unintentional consequence of the limited frequency response of amplifiers, loudspeakers, microphones, or transmission systems.

The end result of all these transformations is a feature vector whose 128 components represent high-pass-filtered perceptual intensity as a function of Bark frequency, in a brief interval around the time point of interest. In order to pair this frequency-domain information with some local temporal information, the psychoacoustic transformation is repeated in adjoining fragments, one immediately preceding and the other immediately following the fragment of interest. These temporal offsets produce a total of three 128-dimensional feature vectors, which are concatenated to form a 384-dimensional feature vector. The time point of interest is advanced through the recording in half-fragment steps of 92.8575 ms, so each time point is incorporated into two different 128-vectors, and each 128-vector is used in three different 384-vectors.

Especially after frequency spreading and duplication across time, these feature vectors contain a great deal of

redundant information. In order to avoid the problem of searching in a very high-dimensional space, the 384-vectors are reduced to 8-vectors by principal components analysis (PCA), a method also applied in a system by Burges et al. [6]. PCA [23] is a standard method for reducing dimensionality by exploiting correlations between dimensions. In 3-space, for example, a set of vectors whose coordinates were perfectly uncorrelated might form a spherical volume of scattering, while a set of vectors whose coordinates were perfectly correlated would form a line. A correlation much greater than 0 but somewhat less than 1 would describe a cylindrical volume, the long axis of which would capture most of the variance in the data. If one wanted to reduce the dimensionality of the data from 3 to 1, PCA could be applied in order to discover this axis of greatest variation, and all of the original 3-vectors could then be projected onto it.

More formally, PCA operates on an  $N$ -dimensional space by computing from a set of training data the  $N \times N$  covariance matrix where element  $(i, j)$  is the covariance of the  $i$ th dimension with the  $j$ th. The eigenvectors of this matrix are the principal components, and for each of these principal components the corresponding eigenvalue describes the amount of variance captured. Reduction from 384 dimensions to a basis of eight dimensions thus can be accomplished by computing the eigenvectors of the  $384 \times 384$  covariance matrix and extracting the eight eigenvectors whose eigenvalues are largest. The transformation matrix from the original 384-space to a variance-maximizing 8-space then consists of these eight vectors, or principal components.

The training data for our PCA consist of all of the 384-dimensional feature vectors computed from nonoverlapping groups of three fragments, that is, from samples at intervals of 557.145 ms. The eight basis vectors computed in this analysis (Fig. 4) account for 46.5% of the variance in our database of 100 000 recordings. Two of these basis vectors capture temporal information, changing sign

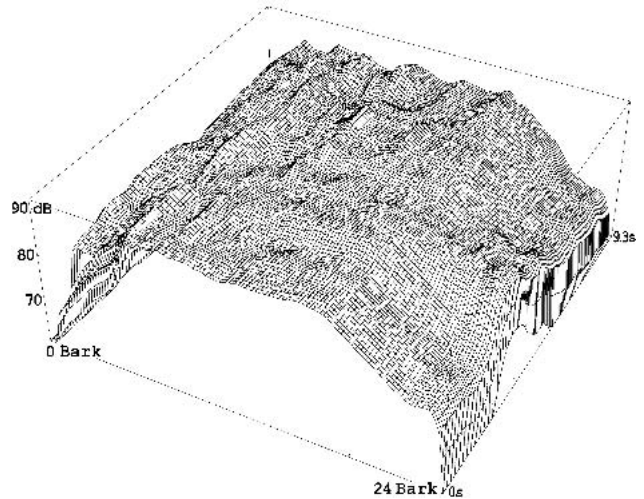


Fig. 2. Evolution of psychoacoustic transformation across successive fragments. Bark frequency is on horizontal axis, loudness on vertical axis, and time on depth axis. In these half-overlapping, 93-ms fragments, psychoacoustic features arise and disappear gradually over time.

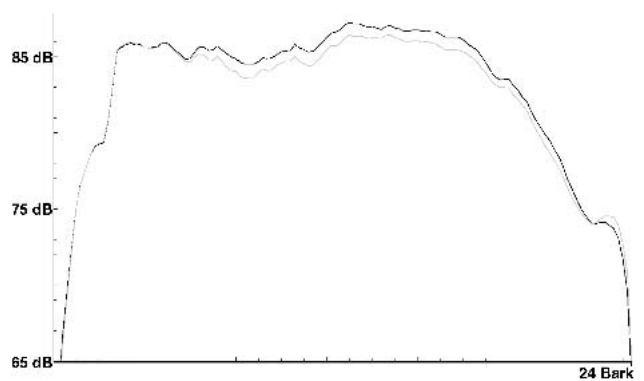


Fig. 3. Comparison of two inputs, identical except for equalization profiles. Note almost exact similarity in local psychoacoustic features and frequency-dependent offset in absolute amplitudes.

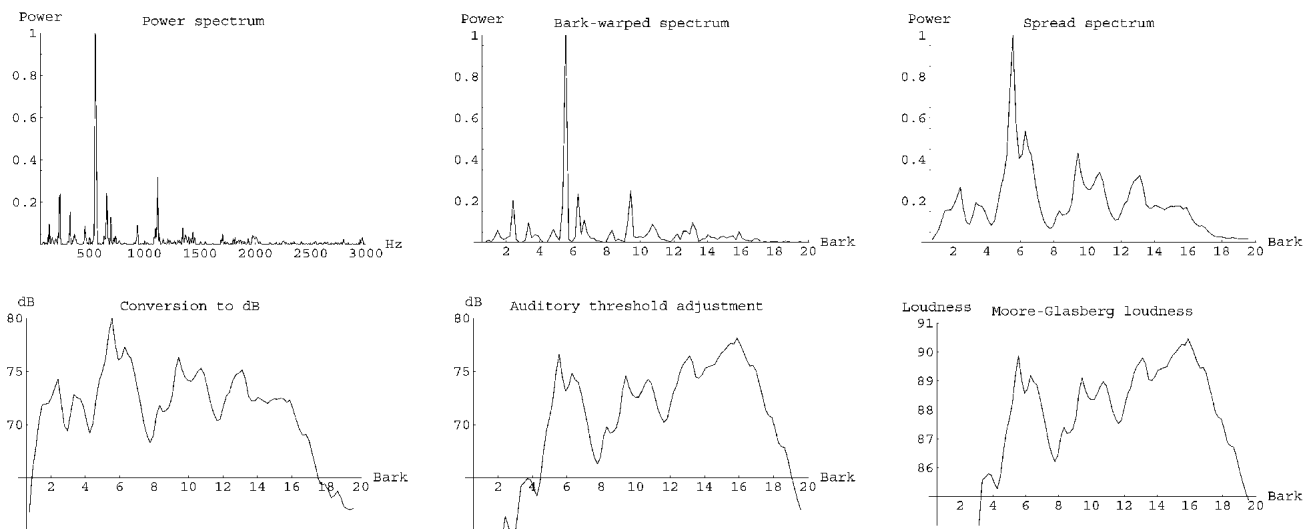


Fig. 1. Steps in psychoacoustic transformation. Power is transformed to a Bark scale and spreading is applied to mimic the mechanical properties of the human cochlea, implementing the psychoacoustic phenomenon of masking. Amplitude of the spread spectrum is converted to dB, auditory threshold is subtracted, and resulting levels are converted to an arbitrarily scaled, dB-like measure of perceptual loudness.

across each group of three successive fragments, and the other six capture frequency-domain information that is fairly constant across successive fragments. Although projecting onto this compact basis preserves slightly less than half of the variance in the total corpus of recordings, our framework of multiple independent comparisons makes the system robust to the resulting increase in match failures. An eight-component basis greatly reduces storage requirements in comparison to larger bases, and even doubling the dimension of the basis would account for only an additional 10.4% of the sample variance (Fig. 5).

### 1.3 Fragment Analysis

Vectors are stored in the database as eight 4-byte IEEE floating-point numbers, with 20 additional bytes associating the vector with its source recording and a temporal offset within that recording. With an average recording length of about 4 min, the 100 000 recordings in the database occupy 12.2 Gbyte of storage. In order to attain high throughput for match queries, this storage is split across seven separate back-end servers, each holding a 1.75-Gbyte slice of the master database. Vectors are quantized into partitions using the generalized Lloyd algorithm [24], [25], and only the partition whose centroid has the closest Euclidean distance to the query vector is searched. One or more front-end index servers accept queries and dispatch each of them asynchronously to the back-end machine that holds the appropriate partition (Fig. 6). In general, the number of back-end machines required for a particular application is a function of both the size of the database (given a limit on memory per single PC bus) and the volume of queries per unit time. These two factors scale somewhat independently: a small, heavily loaded Tuneprint system can split a small amount of memory across a large number of CPUs, whereas a large Tuneprint system with a smaller query volume can concentrate more memory in a smaller number of machines.

The partitioned database can be described by two inversely related measures: entropy and partition error. The entropy of the system is the expected amount of information necessary for its description—in this case the sum, over all partitions, of the probability that a vector will fall

within a given partition times the information contained in such a partitioning:

$$\sum_{\text{partition}} \left[ \frac{|\text{partition}|}{|\text{database}|} \cdot -\log_2 \left( \frac{|\text{partition}|}{|\text{database}|} \right) \right].$$

High entropy decreases processing time. One factor in this decrease is that more partitions allow a database to be spread over many separate back-end servers. More significantly, though, even when many partitions are allocated to a single server, high entropy means that each partition contains fewer search candidates.

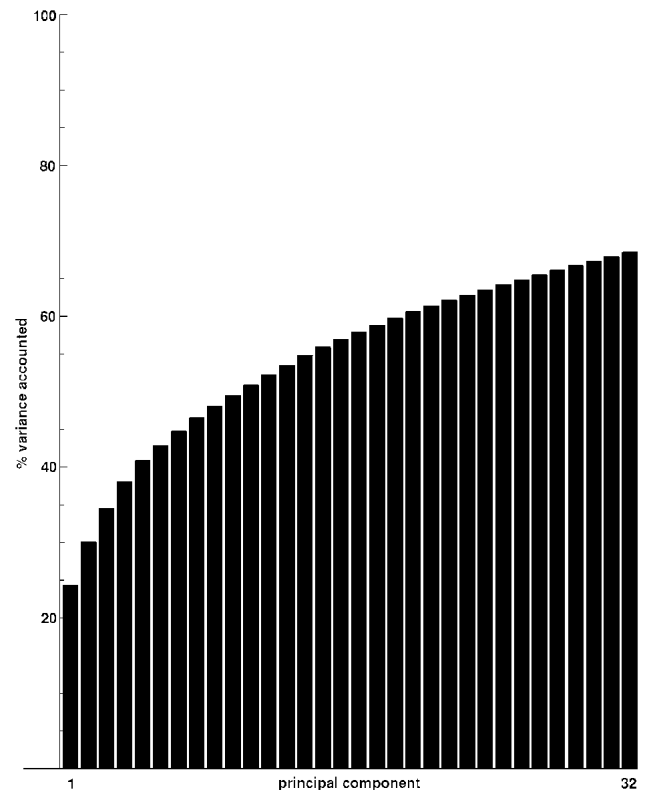


Fig. 5. Variance accounted for as a function of number of principal components preserved. Eight components account for 46.5% of variance. From there, curve increases more gradually.

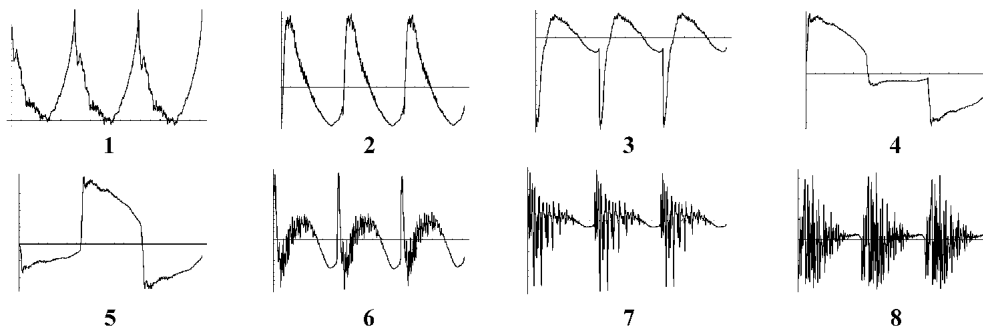


Fig. 4. Basis vectors computed by principal-components analysis. Each input is a concatenation of three 128-dimensional feature vectors in successive fragments; frequency-selective components therefore replicate the same features three times over. Note broad frequency selectivity of components 1 through 3, and time dependence of components 4 (decay and attack) and 5 (beat). Higher order components show a spiky pattern of frequency selectivity, perhaps incorporating information from notes and harmonics. Since basis vectors are normalized to unit length, magnitudes are arbitrary and are not specified here.

The tradeoff is that with high entropy comes a high proportion of incorrectly partitioned queries. In the limiting case, in which a query vector is identical to one of the vectors in the database, of course, the query will necessarily be allocated to the same partition as its best match. However, if the query vector lies some distance from its best match, there is some risk that the query and the best match will fall on different sides of a partition boundary. In such a case the best match will never be identified, because the partition that contains it will not be searched. The optimal partitioning scheme maximizes entropy with the constraint that the rate of query partitioning errors is low enough to produce good matches at a preponderance of time points. The balance between these two factors of high entropy and low partitioning error depends on the average distance between queries and their best matches: again, in the limiting case where this distance is 0, each vector in the database could be its own partition (and the entire problem could be handled using string-matching algorithms).

Within the selected partition we consider only the neighborhood  $N$  of vectors whose Euclidean distance from the query vector  $q$  does not exceed a threshold distance  $d_{limit}$ . The theme behind this computation is to do only as much work as is necessary to produce the desired number of best matches. In general, the algorithm operates by considering successively larger subspaces of the query space. We begin by considering the one-dimensional subspace defined by the first principal component. (Recall that the principal components are the basis in terms of which the query space is represented; projections onto the subspaces defined by these components therefore have zero computational cost, being implemented simply by discarding the higher order dimensions.) To facilitate distance computations within this one-dimensional subspace, the contents of the database partition are maintained in sorted order according to the value of the first principal component. The set  $N_0$  of all vectors that lie within a distance  $\pm d_{limit}$  of the query vector along this principal component can thus be computed with an efficient search.

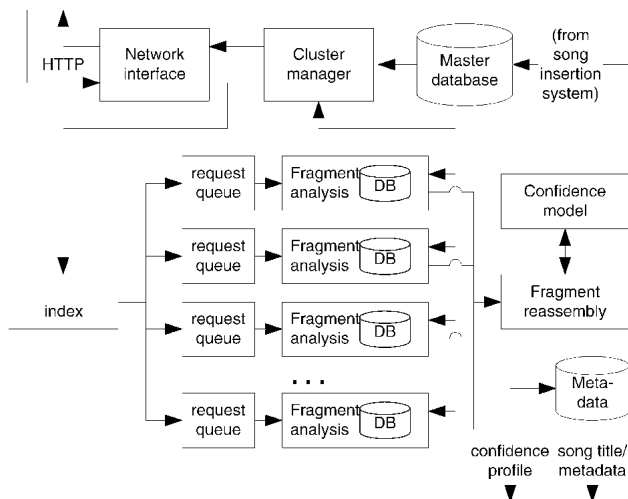


Fig. 6. Block diagram of Tuneprint processing. Queries are dispatched to appropriate back-end server.

Since distance along the principal component is a lower bound for actual distance,  $N \subseteq N_0$ .

In order to exclude those vectors that are in  $N_0$  but not in  $N$ , we organize the elements of  $N_0$  into a heap keyed on lower bound distance from the query vector. A heap, as used in this case, is a binary tree in which the value of every child node is greater than or equal to the value of that child's parent node. The root node in such a tree is termed the "top" of the heap, and it necessarily contains the heap's minimal value. We consider this minimal value, summing into it the squared distances along the next  $\Delta n$  principal components. The addition of these extra dimensions tightens the lower bound on the vector's actual distance from the query. In the case in which this addition causes the value at the top of the heap to grow larger than the distances lower down in the heap, we swap values down the tree in order to reestablish the heap property.

We continue this process, on each iteration incrementing the number of dimensions in the top vector's distance estimate by  $\Delta n$ , and then reestablishing the heap property on the basis of this increased distance estimate. When a vector appears at the top of the heap for which all the dimensions in the query space have been considered, that is, for which the lower bound distance is identical to the actual distance, we identify that vector as the next match, and delete it from the heap. This incremental computation of distances allows us to perform only the computation that is necessary in order to identify the closest matches.

In the case of the eight-dimensional space around which the current implementation is built,  $\Delta n = 7$ , and thus all the higher dimensional distance computation is done at once. However, this incremental algorithm has improved performance in tests with higher dimensionality, and may prove useful if scaling of the system demands an increase in the number of basis vectors. Although the search algorithm without incremental distance computation is  $O[D |N_0| \log(|N_0|)]$  for  $N_0$  vectors in a space of  $D$  dimensions, in practice constant factors are such that this time bound is an improvement on  $O(D|N_0|)$  exhaustive search. The algorithm is specified formally in Fig. 7.

```

Let  $D$  be the dimensionality of the query space;
Let  $q = (q_0 \dots q_{D-1})$  be the query vector within this space;
Let  $k$  be the number of matches desired;
for  $\{v | (v_0 - q_0)^2 \leq d_{limit}^2\}$ 
     $dim(v) := 1$ ;
     $d^2(v) := (v_0 - q_0)^2$ ;
initialize  $N$  as a heap keyed on  $d^2$ ;
MATCH :=  $\emptyset$ ;
do  $[MATCH] \neq k \wedge |N| \neq 0 \rightarrow$ 
    Let  $v$  be the vector at the top of heap  $N$ ;
    if  $dim(v) = D \rightarrow$ 
        MATCH := MATCH  $\cup \{v\}$ ;
        delete  $v$  from  $N$ 
    else  $dim(v) < D \rightarrow$ 
         $d^2(v) := d^2(v) + \sum_{i=dim(v)+\Delta n-1}^{dim(v)+\Delta n-1} (v_i - q_i)^2$ ;
         $dim(v) := dim(v) + \Delta n$ ;
        reestablish  $N$  as a heap on  $d^2$ 
    fi
od
    
```

Fig. 7. Algorithm for incremental search of query neighborhood.

## 1.4 Assembly

Using this procedure we obtain the closest 10 matches for each time step of the query recording. Each of these matching database entries is tagged with its source recording and its temporal offset within that source recording. Assembling the information from these multiple independent matches is a matter of detecting whether a preponderance of them originates from a single recording, with temporal offsets corresponding to their sequence within the query recording.

For each time step in the query, we first scale the results for each recording in the database that has produced more than a single match for this time step. (Singleton match sets are likely spurious—the results of partition errors and of collisions in the database—and are discarded.) If  $M$  is the number of matches originating from a particular matching recording,  $d_i$  is the distance between the query vector and the match vector for the  $i$ th match, and  $(d_{\min}, d_{\max})$  is the interval covering all the  $d_i$ 's, then the score for each match is defined as

$$S = \frac{d_i + d_{\min} - 2d_{\max}}{\left(\sum_{i=0}^{M-1} d_i\right) + Md_{\min} - 2Md_{\max}}$$

This expression scales the best match to 1 and the worst to 1/2, with the intermediate scores being a linear function of the match distance. The resulting scores are pooled across time steps for each matching recording. These match scores over time can be visualized as a two-dimensional plot, with query time on one axis, match time on the other, and the intensity of each pixel corresponding to the score of the match against the query at the corresponding time point. A strong match thus appears as a bright diagonal line (Fig. 8) [16].

The top candidates in terms of number of time steps matched are considered further. Linear regression on the temporal offsets from each match, weighted by the match

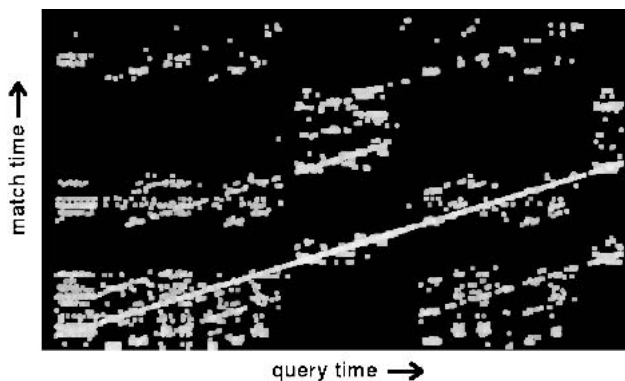


Fig. 8. Graphical representation of a successful match. Horizontal axis is temporal offset within a query recording; vertical axis is temporal offset within a candidate match in database. Intensity of each pixel  $(i, j)$  represents distance between vector representing  $i$ th sample of query recording and vector representing  $j$ th sample of database recording. Thus a perfect match appears as a bright diagonal line, matches against refrains appear as diagonal line segments offset from main line, and absence of a match appears as a dark field.

scores, establishes the slope of the regression line between query time and match time—in other words, the playback speed. In most cases this slope will be 1. However, it is possible for the playback speed to differ slightly from 1, for example, in the case of a broadcaster trying to pack as many tracks as possible into an hour of air time. In applications in which the playback speed is known to be 1, this regression step can be skipped.

Linear regression could also be applied to estimate the intercept, that is, the length of the recording's initial silence or the amount by which the recording has been truncated at its beginning. Such an estimate, however, would be perturbed by the repetition of themes and refrains, which produce short sequences of matches with very different temporal offsets (see Fig. 8). In order to avoid such perturbation, we estimate the intercept by sliding the regression line through the match plot and computing the total number of matches within a five-point window centered on the regression line. The intercept that maximizes this local match total is taken as the actual temporal offset, and the maximal total itself is the match score of the recording as a whole. All the recordings under consideration are then ranked in order of this match score.

## 2 PERFORMANCE

For a preliminary evaluation of Tuneprint's performance on real-world data, a test corpus was created of randomly selected recordings that were publicly available as MPEG-1 layer 3 files on a peer-to-peer network. In cases in which listening tests indicated that one of these downloaded recordings was not in our database, or was a duplicate of a recording already in the test set, the recording in question was deleted from the test set. These deletions left a test set of 73 unique recordings. In all 73 cases, Tuneprint identified the test recording correctly as the database entry with the greatest match score.

The system used to conduct this test split 1000 database partitions between 14 server processes, two on each of seven physical servers. Each physical server had two 1.0-GHz Athlon processors and 2 Gbyte of RAM. Since each partition is a different size, a simple, greedy method was used to balance the total number of vectors assigned to each server process, considering each partition in turn from largest to smallest and assigning it to the server process with the smallest total number of vectors so far. This system computed matches for an entire recording against its database of size 100 000 in approximately 1.5 s of CPU time. This figure includes the client's time spent generating query vectors from the audio input, the front-end database server's time spent dispatching queries, and the back-end servers' time spent matching the queries.

The entropy of the partitioned database was 8.45, equivalent to a collection of 349 equally likely classifications. Fig. 9 shows the linear relationship, within this test database, between the distance of a query vector from its match target and the fraction of queries lost to partition errors. In our test system, about 25% of queries were lost to partition errors. However, this high error rate is well tolerated by Tuneprint's system of multiple independent

comparisons: as long as there are enough successful queries to raise the score of the matching database entry above those of nonmatching entries, the input will be correctly identified. An increase in partition error simply produces a corresponding increase in the length of input required for reliable identification. For example, 150 fragments at 50% loss will produce identifications as reliable as 100 fragments at 25% loss, since on average both scenarios yield the same number of successful queries, namely, 75. Although the relationship between match scores and identification rate has not yet been systematically explored, our experience is that the top match can be reliably detected if it exceeds the next closest match by a score of 8 or more. (Since matches at individual queries have values between  $1/2$  and 1, this threshold corresponds to matching at 8 to 16 individual fragments. Recall that the frames from which each fragment is produced are spaced at intervals of 92.8575 ms. Therefore this threshold corresponds to an input length of between about 0.75 and 1.5 s, neglecting the effect of temporal autocorrelation between frames. These figures match our experiences with inputs consisting of short clips, where identification becomes unreliable for input lengths below about 1 s.)

### 3 FURTHER WORK

The current Tuneprint system is a work in progress, several aspects of which deserve further research. Some of these issues are points that will become important as the system scales up beyond the current 100 000 recordings. Others are possibilities for improvement regardless of scale.

With large databases, the question of what constitutes an acceptable test set becomes increasingly important. Tuneprint is meant to be applied to recordings that have

undergone lossy compression and other subtle distortions as are commonly found on file-sharing networks, and is designed to implement an identity criterion of equivalence to a human listener. Any validation of Tuneprint's output must therefore depend on a sampling of recordings representative of those found on file-sharing networks, and on the independent evaluations of human listeners which, for this purpose of validation, cannot be automated. This requirement for listening tests makes expansion of the test set very labor intensive and is the reason for the current test set's small size. More significant than the size of the test set, though, is the size of the database against which the test recordings are being matched. It is the database size that determines the system's liability to collisions, and hence to false positives and to match failures. Since the elements of the test set are randomly selected, each element constitutes an independent test of the database. Although the further development of Tuneprint would benefit from more detailed and larger scale testing than was possible during the period of Tuneprint's commercial funding, the present result with a database of size 100 000 is an indicator of the promise of Tuneprint's methods.

Partly because of the small size of Tuneprint's test set, a detailed, quantitative comparison with the performance of other systems remains an open question. Wang [17] independently implemented a temporal consistency measure similar to Tuneprint's, though without full psychoacoustic modeling at the input stage; the experimental performance of this system has not been reported. Using the vector quantization method of Allamanche et al. [9], Hellmuth et al. have implemented an advanced classifier [10] that makes use of the information inherent in the temporal sequence of feature vectors, though the exact nature of this method is left unspecified. An alternative to vector quantization methods is robust hashing, in which feature vec-

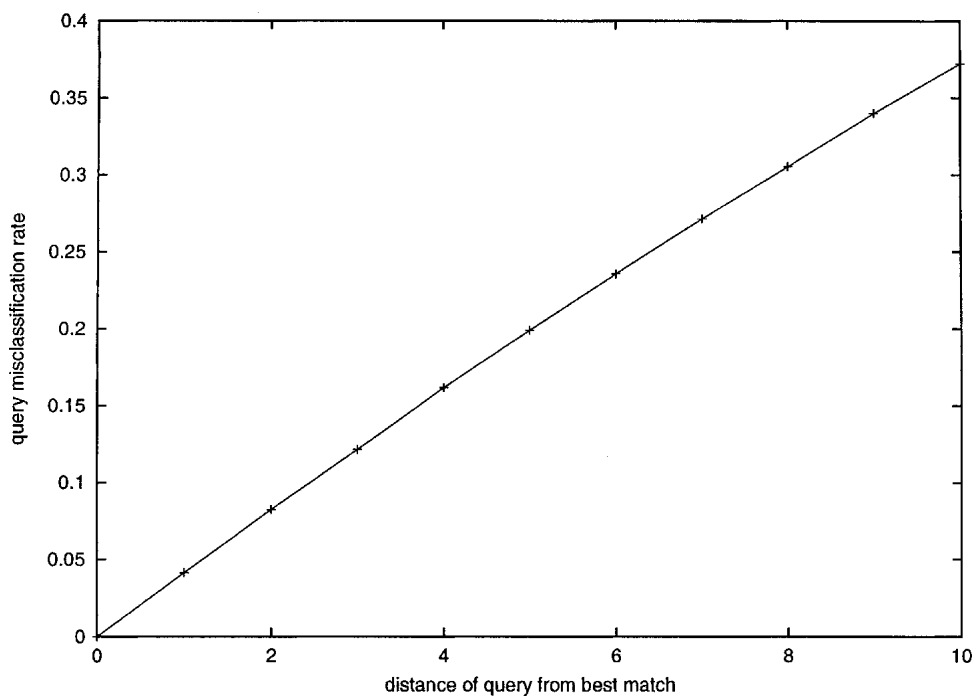


Fig. 9. Rate of query misclassification as a function of distance between query and target. Note linearity of relationship.

tors are discretized by simple thresholding of their components, and the search space for approximate matches is then constrained to only those database recordings of which one or more frames produced an exact match in this discretized domain [11]. Although the distribution of within-recordings hash errors in a database of size 10 000 using this method theoretically predicts a very low rate of false positives with a high rate of identification [12], the between-recordings error rate has yet to be assayed experimentally in a large database. An important feature of systems based on multiple independent comparisons is that the method of analysis within time points is separable from the method of evaluation for consistency across time points. This separability raises the possibility of plugging any within-points method into an alternative across-points method—for example, Tuneprint's psychoacoustically based model could be integrated with a robust hashing system.

Another key question regarding scaling is how the growth of the database may increase the likelihood of collision, that is, a situation in which two items in the database map so close together in Tuneprint's search space that discrimination between them is degraded or impossible. As Tuneprint matches first at the level of isolated fragments and then at the level of whole recordings, there are two senses in which collision can be considered. First, collisions might occur between individual fragments of different recordings. Such collisions would affect matching of the recording as a whole in the same way that partitioning error does. Tuneprint's redundant strategy of multiple independent matching makes it robust to collisions at the fragment level in the same way that it is robust to partitioning error—as long as there are significantly many successful matches, failed matches do not affect the identification [15]–[17]. Although in tests to date any small effect of such collisions has been swamped by the effect of partitioning error, a high rate of collisions could be expected to mimic the effect of partitioning error, increasing the length of input required for reliable identification.

Second, one could imagine a collision involving very strong matching to more than one recording in the database. Although we have observed such occurrences during our tests, all have turned out to involve cases in which a recording loaded into the database from a peer-to-peer network had been mislabeled and was actually a copy of another matching recording already present in the database. We have never observed a true collision at the level of whole recordings, and this absence of collision may perhaps be expected given the length of a typical recording and the number of dimensions within which it can vary as it evolves through time.

Scaling certainly can be expected to figure into the tradeoff between entropy and partitioning error. It remains undetermined what degree of partitioning would be optimal even at the database's current size. The limit in which partitioning errors begin to affect accuracy has not yet been reached, and the current stopping point for the number of partitions is somewhat arbitrary. Assuming that the rate of partitioning errors remains a linear function of the distance between the query and its best match (Fig. 9), a

database partitioning can be characterized by the slope of this partitioning-error function in combination with the entropy figure. The total expected CPU time per identification is the product of the expected number of queries (a function of the partitioning-error rate) and the expected CPU time per query (a function of entropy). It likely will be possible to develop an optimization procedure that applies these descriptive statistics to find the partitioning scheme that minimizes the expected CPU time per identification.

Perhaps the most compelling question regarding partitioning is that of how much information needs to be preserved within a single partition. Vector quantization yields an efficiency of time, arising from the restriction of the search space to a particular partition, and an efficiency of space, arising from the lossy coding of individual vectors in terms of the partition to which each maps. Currently Tuneprint takes advantage only of the temporal efficiency, applying vector quantization to select a particular database partition to search. The potential spatial efficiency is not realized, since the original, unquantized vectors are preserved for use in nearest-neighbor matching within the selected partition. A database with very high entropy might offer the potential for eliminating the query-to-match distance measure within a partition, and instead treating all elements of the partition as equally good matches. The winning match at the level of whole recordings could then be determined by consistency of matching as in the current model. Such an ability to discard the original vectors would offer large savings in space, by eliminating the bulk of the database, and in time, by eliminating the demand for nearest-neighbor matching in a high-dimensional space. We have implemented a prototype of such a system which compresses the entire database into 3 Gbyte, runs on a single CPU, and, when successful, identifies a recording in only 110 ms of processing time on average.

Variations on this high-entropy strategy also are possible. One optimization might involve overlapping partitions, that is, allowing database vectors that lie near partition boundaries to be included in more than one partition. Conversely, query vectors that lie near partition boundaries could be made to trigger a search of multiple partitions in the database. Either of these strategies would decrease the rate of partitioning error, at the expense of a modest increase in search time. One can also imagine a two-pass system in which distance measures are computed only for those match candidates that show consistent matching over time at the level of partitions.

Currently the query is sampled at constant intervals throughout its length. Higher confidence in matching likely can be achieved by varying this sampling period, both as a function of overall match confidence (more sampling of difficult-to-identify recordings) and as a function of the temporal derivative of the psychoacoustic function (more sampling in the time intervals surrounding abrupt changes). Landmarking of the input recording is one way of implementing increased sampling at intervals of abrupt change, and it would be interesting to compare the performance of landmarking based on simple acoustic properties

[17] with landmarking based on psychoacoustic properties. Although in general the psychoacoustic output varies slowly over time, the presence of two principal components that encode time-varying information from neighboring fragments is an indication that abrupt changes contain much useful information.

Although dimensionality reduction by principal components analysis is an expedient strategy, it may not be the best way to form a basis. In particular, the roles of the two time-varying components extracted by PCA have yet to be ascertained. In an environment in which queries may be offset from their best matches by up to half the length of a fragment, an inclusion of time-varying components may simply add noise. In addition, the construction of a space based on independent component analysis (ICA) [26] has not yet been explored, and should be.

Although Tuneprint has performed well in preliminary tests, the effects of distortion on Tuneprint's performance have not yet been explored systematically. Since our test set was selected at random from recordings publicly available on peer-to-peer networks, its audio quality was representative of this population. Our statistics indicate that only about 1% of these recordings are encoded at bit rates less than 128 kbits/s. In addition, our bandwidth of 253 to 12 500 Hz ranges four times as high as that available in telephone transmission. Testing and retuning for low-bit-rate encodings, for signals varying in playback speed, and for signals varying in bandwidth and other equalization properties were under way with positive early results, but unfortunately could not be completed within the period of Tuneprint's commercial funding. The outlook for low-bandwidth processing seems particularly positive since most of the amplitude of the basis vectors (see Fig. 4) is contained in the lower Bark frequencies.

It is worth noting that audio is not the only medium to which our general strategy of multiple independent matching of fragments might apply. Any medium that can be reduced to an array of features, either over time or over spatial dimensions, can be subjected to fragmentation over these dimensions. A fragment analysis of images, for instance, might compare features that are independent of translation, rotation, and scaling, and the corresponding assembly function might compute affine transformations between the query image and matching portions of the database images.

We have described a system for content-based identification of audio recordings, built around a strategy of multiple independent matching. Using a psychoacoustically based representation, this system matches inputs according to a criterion meant to model that of a human listener. Its identifications are robust to truncation, and work well with the lossy compression commonly found in recordings on peer-to-peer networks. In addition, the system is designed to be robust to modest variations in equalization and playback speed. For equalization in particular, we found that high-pass filtering of the output of the psychoacoustic transform (see Fig. 3) was very effective in maintaining performance. Successful operation on a database of 100 000 recordings bodes well for further scaling.

## 4 ACKNOWLEDGMENT

Profound thanks are due to our colleagues Daren Gill, Martin Stiaszny, Amittai Axelrod, Josh Pollack, Jennifer Chung, and Lex Nemzer, without whose many day and night hours the Tuneprint system could never have been implemented. In addition, we wish to acknowledge Sage Hill Partners, who funded the development of Tuneprint.

## 5 REFERENCES

- [1] A. Pikrakis, S. Theodoridis, and D. Kamarotos, "Recognition of Isolated Musical Patterns in the Context of Greek Traditional Music," presented at the IEEE Int. Conf. on Electronics, Circuits, and Systems, Rhodes, Greece, 1996 Oct. 13–16.
- [2] J. T. Foote, "Content-Based Retrieval of Music and Audio," in C. C. J. Kuo, Ed., *Multimedia Storage and Archiving Systems II, Proc. SPIE*, vol. 3229, pp. 138–147 (1997).
- [3] A. Pikrakis, S. Theodoridis, and D. Kamarotos, "Recognition of Isolated Musical Patterns Using Discrete Observation Hidden Markov Models," presented at the European Signal Processing Conf., Rhodes, Greece, 1998 Sept. 8–11.
- [4] D. Pye, "Content-Based Methods for the Management of Digital Music," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing* (Istanbul, Turkey, 2000), vol. 4, pp. 2437–2440.
- [5] C. Papaodysseus, G. Roussopoulos, D. Fragoulis, T. Panagopoulos, and C. Alexiou, "A New Approach to the Automatic Recognition of Musical Recordings," *J. Audio Eng. Soc.*, vol. 49, pp. 23–35 (2001 Jan./Feb.).
- [6] C. J. C. Burges, J. C. Platt, and S. Jana, "Distortion Discriminant Analysis for Audio Fingerprinting," *IEEE Trans. Speech Audio Process.*, vol. 11, pp. 165–174 (2003).
- [7] S. Sukittanon and L. E. Atlas, "Modulation Frequency Features for Audio Fingerprinting," presented at the IEEE Int. Conf. on Acoustics, Speech, and Signal Processing, Orlando, FL, 2002 May 13–17.
- [8] E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-Based Classification, Search, and Retrieval of Audio," *IEEE Multimedia*, vol. 3, pp. 27–36 (1996).
- [9] E. Allamanche, J. Herre, O. Hellmuth, B. Fröba, and M. Cremer, "AudioID: Towards Content-Based Identification of Audio Material," presented at the 110th Convention of the Audio Engineering Society, *J. Audio Eng. Soc. (Abstracts)*, vol. 49, p. 542 (2001 June), convention paper 5380.
- [10] O. Hellmuth, E. Allamanche, J. Herre, T. Kastner, M. Cremer, and W. Hirsch, "Advanced Audio Identification Using MPEG-7 Content Description," presented at the 111th Convention of the Audio Engineering Society, *J. Audio Eng. Soc. (Abstracts)*, vol. 49, pp. 1227–1228 (2001 Dec.), convention paper 5463.
- [11] J. Haitzma, T. Kalker, J. Oostveen, "Robust Audio Hashing for Content Identification," presented at the International Workshop on Content-Based Multimedia Indexing, Brescia, Italy, 2001 Sept. 19–21.

[12] J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System," presented at the 3rd Int. Conf. on Music Information Retrieval, Paris, France, 2002 Oct. 13–17.

[13] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson, and S. J. Cunningham, "Towards the Digital Music Library: Tune Retrieval from Acoustic Input," in *Proc. 1st ACM Int. Conf. on Digital Libraries* (Bethesda, MD, 1996), pp. 11–18.

[14] J. J. Hopfield and C. D. Brody, "What Is a Moment? Transient Synchrony as a Collective Mechanism for Spatiotemporal Integration," *Proc. Nat. Acad. Sci. USA*, vol. 98, pp. 1282–1287 (2000).

[15] G. Schmidt, "Tuneprint Audio Fingerprinting Technology: Technical Overview Version 1.1," paper prepared in response to Recording Industry Association of America/IFPI request for information (2001 June 6).

[16] G. Schmidt, "Tuneprint Fingerprinting Technology," presented to the Recording Industry Association of America, Washington, DC (2001 Sept. 5).

[17] A. Wang, "System and Methods for Recognizing Sound and Music Signals in High Noise and Distortion," International Patent Publ. WO 02/11123 A2 (2002).

[18] E. Zwicker and H. Fastl, *Psycho-acoustics: Facts and Models*, 2nd ed. (Springer, Berlin, 1999).

[19] K. Brandenburg and G. Stoll, "ISO/MPEG-1 Audio: A Generic Standard for Coding of High-Quality Digital Audio," *J. Audio Eng. Soc.*, vol. 42, pp. 780–792 (1994 Oct.).

[20] J. G. Beerends and J. A. Stemerding, "A Perceptual Audio Quality Measure Based on a Psychoacoustic Sound Representation," *J. Audio Eng. Soc.*, vol. 40, pp. 963–978 (1992 Dec.).

[21] ISO 389-7 "Acoustics—Reference Zero for the Calibration of Audiometric Equipment—Part 7: Reference Threshold of Hearing under Free-Field and Diffuse-Field Listening Conditions," International Organization for Standardization, Geneva, Switzerland (1996).

[22] B. C. J. Moore, R. W. Peters, and B. R. Glasberg, "Detection of Decrements and Increments in Sinusoids at High Overall Levels," *J. Acoust. Soc. Am.*, vol. 99, pp. 3669–3677 (1996).

[23] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. (Springer, New York, 2002).

[24] Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. Commun.*, vol. 28, pp. 84–95 (1980).

[25] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. Inform. Theory*, vol. 28, pp. 129–137 (1982).

[26] A. J. Bell and T. J. Sejnowski, "An Information-Maximization Approach to Blind Separation and Blind Deconvolution," *Neural Comput.*, vol. 7, pp. 1004–1034 (1995).

## THE AUTHORS



G. Schmidt

Geoff Schmidt was born in Branson, MO, in 1980. He left undergraduate studies at the Massachusetts Institute of Technology in Cambridge, MA, after one term to pursue entrepreneurial interests. Working alone, he developed the Tuneprint framework in mid-2000 and in 2001 secured venture capital funding from Sage Hill Partners in Cambridge to scale up and commercialize the system. Tuneprint Corporation ceased operations in 2002.

Mr. Schmidt's previous work on output-sensitive visible surface determination algorithms for 3-D rendering has won awards from the International Science and Engineering Fair, the U.S. Junior Science and Humanities Symposium, and other research competitions. He is now a commercial research consultant, currently performing machine vision research at Intellivid, a Cambridge startup company.

Mr. Schmidt's personal interests include meditation, teaching, and politics. He recently served as campaign manager for Matt DeBergalis's Cambridge City Council campaign, where in addition to day-to-day management he designed a successful direct mail effort. He plans to pursue

a career in machine learning or programming language design. E-mail [gschmidt@mit.edu](mailto:gschmidt@mit.edu).



Matthew Belmonte studied English literature and computer science as an undergraduate, developing an interest in the processing of formal and natural languages. Moving from artificial to biological computing systems, he applied this computational interest to problems in cognitive neuroscience. He is the author of several papers on the neurophysiology of attention and perception in normal and autistic populations and on computational methods for statistical analysis of biophysical time series. He was a research scientist at Tuneprint Corporation, the architect of Tuneprint's psychoacoustic model, and a major contributor to the technical description of the Tuneprint system. He left the United States in 2002 and currently works in functional magnetic resonance imaging at the University of Cambridge Autism Research Centre in the UK. E-mail: [belmonte@mit.edu](mailto:belmonte@mit.edu).