MUSART: Music Retrieval Via Aural Queries

William P. Birmingham, Roger B. Dannenberg, Gregory H. Wakefield, Mark Bartsch, David Bykowski, Dominic Mazzoni, Colin Meek, Maureen Mellody, William Rand

University of Michigan
128 ATL Building
1101 Beal Avenue
Ann Arbor, MI 48109-2110
(734) 936-1590
wpb@eecs.umich.edu

Carnegie Mellon University
School of Computer Science
Pittsburgh, PA 15213
(412) 268-3827
rbd@cs.cmu.edu

ABSTRACT

MUSART is a research project developing and studying new techniques for music information retrieval. The MUSART architecture uses a variety of representations to support multiple search modes. Progress is reported on the use of Markov modeling, melodic contour, and phonetic streams for music retrieval. To enable large-scale databases and more advanced searches, musical abstraction is studied. The MME subsystem performs theme extraction, and two other analysis systems are described that discover structure in audio representations of music. Theme extraction and structure analysis promise to improve search quality and support better browsing and "audio thumbnailing." Integration of these components within a single architecture will enable scientific comparison of different techniques and, ultimately, their use in combination for improved performance and functionality.

1. INTRODUCTION

We are integrating components for music search and retrieval into a comprehensive architecture called MUSART, an acronym for *MUSic Analysis and Retrieval Technology*. Like several other music-retrieval systems, MUSART takes as input an aural query, which is typically a theme, hook, or riff, of the piece for which the user is searching. Unlike other systems, however, MUSART automatically builds a thematic index of the pieces in its database. Since users generally remember the theme of a piece of music, and the theme can occur anywhere in a piece, indexing by theme can greatly improve both the precision and recall of the retrieval system.

Moreover, MUSART uses a variety of representations to support multiple search modes. These representations run from a Markov model, to phonetic streams, to strings. This allows us, for example, to easily compute approximate matches and to search based on stylistic similarity or the lyrics in a popular song. Our representation can capture harmony and rhythm, should the user decide to query based on harmonic progression or rhythmic pattern, or both, in addition to or in place of melody.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page.

The current version of the system contains hundreds of pieces from different Western genres (all are tonal pieces). From these pieces, we have automatically induced about 2000 themes. MUSART is able to effectively retrieve pieces from either the theme or full piece databases. The system is relatively robust against queries that contain some classes of errors (e.g., rhythmic changes). We measure performance by rank (the *target* piece is in the top ten pieces retrieved), yielding measures from 100%[1] for queries without errors and degrading from there based on the type of error in the query.

In this paper, we describe the MUSART system architecture[2]. The discussion concentrates mostly on searching themes, as we believe this will be the primary method most users will employ. We will mention, however, extensions to full pieces and other music-analysis strategies. We begin by placing our work in the context of current systems.

2. RELATED RESEARCH

There are varieties of approaches described in the database and information-retrieval (IR) literature on retrieval of music. Some of these approaches, such as Variations [1], are primarily based on retrieving either musical scores or sound recordings using traditional categorization schemes, where the musical items are treated in much the same way as text-based media.

A number of other systems [2-8] have focused on sound and MIDI [9] input. These systems generally take as input a melody that is "hummed" or played on some type of musical input device, such as a MIDI keyboard. The hummed melodies are converted to text strings, usually with a representation of intervallic distance or simply relative pitch contour [10]. For the melody "Happy Birthday," a user may hum the pitches "G G A G C B," which may then be more simply categorized as ascending (U), descending (D), or the same (S) to yield the pitch contour "S U D U D." A commonly used variation to the SUD approach is to divide jumps into large (U or D) and small (u or d), where large is, for example, a jump greater than a minor 3rd. This SUuDd alphabet provides more information than a string composed from the SUD alphabet, but does not substantially change the retrieval process.

[1] That is, in 100% of the cases the target was in the top-ten rank.

[2] See musen.engin.umich.edu for more information on the MUSART project and related projects.

While hummed input is a natural approach, there are some problems with using it as the sole input. Pitch contour and related representations (such using actual or modified pitch normalized to some key) are not necessarily unique: the hummed input is sometimes fraught with a number of distortions inherent in the way that humans remember and produce (sing) melodies [6]. Such distortions include raising or lowering the pitch of various notes, "going flat" over time, losing the beat, or humming "off key." To account for these distortions, researchers treat the string as imprecise. In other words, regions of uncertainty are added. The retrieval process on imprecise strings employs a string-matching [11], N-gram [12], or other algorithm where the input string is matched against abstractions stored in the database.

While this approach has clearly led to some impressive results, we argue that the "string approach" is fundamentally limited for the following reasons:

- The assumption that a melodic fragment exists and forms a suitable search key is not always true for some types of music such as rap, electronic dance music, and even some modern orchestral music. Users may want to search for non-melodic attributes.

- Given a large, but not unrealistic, corpus of music, pitch contour or intervallic representations will not uniquely identify pieces. In other words, there will be a great many pieces with similar string representations; this problem is exacerbated if the strings are modeled as imprecise.

- In some genres, harmony, homophony or polyphony may be predominant musical features. Representing any of these as simple strings is fraught with significant problems. For example, would a two-voice piece be represented as two "concurrent" strings?

We are not arguing against melodic search *per se*; we only want to recognize that current practice must be augmented with new techniques to achieve better performance and richer functionality. For example, current research suggests using genre to narrow searches, but in many cases users may want to search for orchestration, themes, or rhythms that span many genres. Using genre to narrow searches is similar to MARC-record searching in that pre-established categories are selected using traditional database and query techniques. We look to implement more general and more powerful searching techniques.

Several researchers have described systems based on various non-string methods to represent and classify music [8, 13, 14]. These projects have not yet demonstrated how to integrate general query mechanisms and methods for returning results. Nor do these systems integrate abstraction with a search mechanism.

Retrieval by melody depends on the "hook" of a piece of music. The hook is usually what a person uses as a query when humming. The problem is that although the hook may be contained in the opening melodic line of a song, often it is not. For example, it is common in popular songs for a familiar chorus (such as "Take me out to the ballgame…" by Jack Norworth & Albert Von Tilzer) to follow an unfamiliar verse (such as "Nelly Kelly loved baseball games…," the actual opening line from the 1927 version of "Take Me Out to the Ballgame"). In classical music, the main theme (or melody) may not be stated for several measures following the start of the piece, or it may be disguised in a variation, and the variation may be better known than the main theme.

Thus, there is a significant problem in determining what part of a melody, or even which melody, should be indexed. Some abstraction of a piece is clearly needed, as many pieces of music are too long to be reasonably searched. Faced with these problems, music librarians have developed thematic indexes which highlight significant themese in staff notation, thus preserving major musical features (harmony, rhythm, etc.) [13].

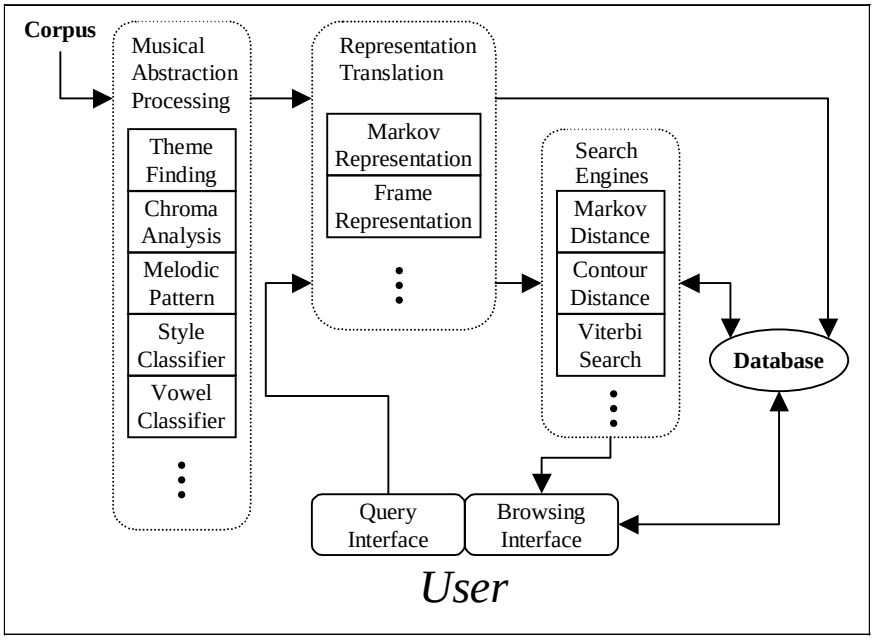


Figure 1. MUSART Architecture.

We know of no other researchers who are addressing the problem of abstraction using automated mechanisms.

3. MUSART ARCHITECTURE

This section illustrates the conceptual organization of our system (see Figure 1). The corpus is preprocessed using a collection of tools to build abstract representations of the music (i.e., our data). The extracted information is then translated to multiple representations. Queries are also translated, and a search engine is selected to search the database. The important idea of this framework is that various modules and representations can work together in parallel or in sequence to achieve more refined searches than any single search technique can offer. We can also compare different techniques on the same set of data.

With this architecture, the user interface can offer a rich set of options to the user. Because music is abstracted in various ways, users can explore musical spaces along many dimensions. In addition, abstract representations lend themselves to music generation and synthesis. We have just begun to explore this possibility, but this promises to be a powerful tool for users to refine queries and to explore different musical dimensions.

So far, we have implemented several new techniques for searching, and we have investigated several techniques for musical abstraction. These are described in the following sections. We are in the process of integrating these subsystems to form a more integrated whole.

4. MARKOV REPRESENTATION AND THE CONCURRENCY

One way we model music is as a stochastic process, where we cast a musical performance as a Markov model. Recently, we have experimented with Markov models where we use simple state features of pitch (or concurrent set of pitches) and duration [14]. The transition table is the likelihood of going from one state to another, where the priors are determined by counting transitions occurring in some set of music (e.g., the piano concerti of Mozart).

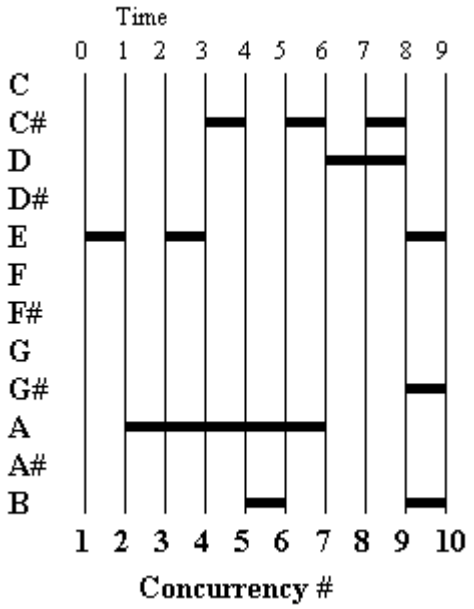


Figure 2: Piano roll for *Wilson's Wilde*, mm. 1 – 4.

State #	Notes in State
1	E
2	A
3	E, A
4	C#, A
5	A, B
6	C#, A
7	D
8	C#, D
9	E, G#, B
10	REST

Table 1: Concurrencies for Wilson's Wilde, mm. 1-4.

We define the states of the Markov model as *concurrencies*. A concurrency consists of all the notes that are sounding at the same time. These notes need not all have the same duration, onset, or offset. Hence, a different concurrency exists at each point a new note begins or terminates in a piece. For the purposes of simplification, the notes in a concurrency are modeled by pitch class, thereby ignoring octave information. By representing concurrencies in this manner, we are able to store and use them as 12-tuple bit vectors. It is easy to extend the concurrency to include octave information, simply by increasing the size of the bit vector. For example, a two-octave range is represented by a 24-tuple and corresponding bit vector.

The concurrency can represent up to twelve simultaneous pitches (every pitch class); elements of the harmony or polyphony are naturally represented. Moreover, this represent is well suited for automated harmonic analysis [15].

We use Figure 2 and Table 1 as examples of a set of concurrencies and the corresponding state and transition tables for the Markov model representing the piece. Figure 2 shows the piano-roll notation for *Wilson's Wilde*, an anonymous 16th-century work. Table 1 shows the corresponding concurrencies (given in the table as Markov states) based solely on this excerpt.

Although relatively simple, the Markov representation has several interesting properties that allow us to use it as one of the primary representations in our system. First, our results from inducing Markov models from the pieces in our corpus indicate that composers occupy (mostly) unique regions in the state space implied by the Markov model. Secondly, we found that the transition matrices are relatively sparse (more so for some works than others). Thirdly, we can impose an order on the states implied by the Markov model.

The ordering is formed as follows: we quantize note duration to some minimum (e.g., 16th-note) and use a pitch-class vector where, for example, <100000000000> represents the pitch class C as the single pitch sounding, <110000000000> represents the pitch classes C and C#/D-flat sounding concurrently, and so forth. A state is represented by a duple (duration, pitch vector). We can simply order the state space as follows: (16th-note, <100000000000>), (8th-note, <100000000000>), … (whole note,

<111111111111>), where durations are ordering by increasing powers of 2 of a 16th-note, and the maximum duration is a whole note.[3]

Because of these three properties, we can assess similarity among pieces of music, or even composers using a variety of techniques. Initial results indicate a strong correspondence between similarity as computed by the MUSART system and educated musical opinion. Consider that once the query is converted to a Markov chain, it can be easily correlated with pieces in the database (See Section 6.) While at worst case this is a linear-time operation, the correlation operation is fast and, with clever indexing, we can significantly reduce this time. Finally, the induction of the Markov model and correlation computation can be done off line.

5. Thematic Abstraction

We are interested in extracting the major themes from a musical piece: recognizing patterns and motives in the music that a human listener would most likely retain. Extracting themes is an important problem to solve. In addition to aiding music librarians and archivists, exploiting musical themes is key to developing efficient music retrieval systems. The reasons for this are twofold. First, it appears that themes are a highly attractive way to query a music-retrieval system. Second, because themes are much smaller and less redundant than the full piece, by searching a database of themes rather than full pieces, we simultaneously get faster retrieval (by searching a smaller space) and get increased relevancy. Relevancy is increased as only crucial elements, variously named motives, themes, melodies or hooks are searched, thereby reducing the chance that less important, but frequently occurring, elements are assigned undue relevancy.

Our theme abstraction subsystem, MME [16], exploits redundancy that is found in music. Thus, by breaking up a piece into note sequences and seeing how often these sequences repeat, we identify the themes. Breaking up a piece involves examining all note sequence lengths of one to some constant. Moreover, because of the problems listed earlier, we must examine the entire piece and all voices. This leads to very large numbers of sequences, thus we must use a very efficient algorithm to compare these sequences.

Once repeating sequences have been identified, we must further characterize them with respect to various perceptually important features in order to evaluate their thematic value. It has been noted, for instance, that the frequency of a pattern is a stronger indication of thematic importance than pattern register. We implement hill-climbing techniques to learn weights across features, i.e., MME learns relative to a training set the relative importance of the features it uses. The resulting evaluation function is then used to rate the sequence patterns we uncover in a piece. A greedy algorithm is used to identify a subset of the piece, consisting of some pre-determined number of note events, containing top-ranked patterns.

Our formal evaluation of MME on over 30 training trials, with 30-piece training and test sets randomly selected across 60 pieces for each trial, MME returns Barlow's *A Dictionary of Musical Themes* [17] "1st theme" in 98.7% of cases (see Figure 4). Figure 3 shows sample output from MME, two slightly different versions of the passage Barlow identifies as the "1st theme".

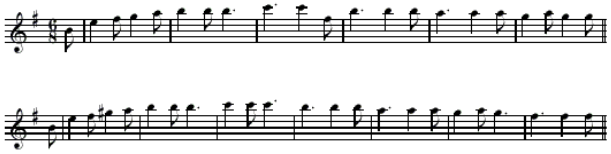


Figure 3: Sample MME output, Smetana's *Moldau.*

Because of the large number of patterns that MME may find in a complex piece of music, it must be computationally efficient. The system's overall complexity is $\Theta(m^3 n^2)$ time, where m is the maximum pattern length under consideration, and n is the number of note events in the input piece. In practice, however, we observe sub-linear performance, and reasonable running times on even the largest input pieces.

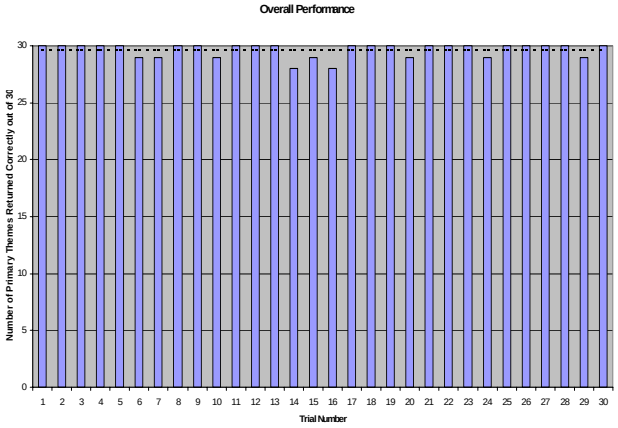


Figure 4: MME test results.

6. Retrieval Methods

The pieces in the MusArts' database are converted by MME into a set of themes, where up to a certain number of themes are identified for each piece. These themes are then converted to the Markov models described in Section 4. Thus, the "Markov Distance" retrieval engine takes a query, converts it to a Markov model, and then computes a variety of correlations measures between the query model and all "theme" models [18].

Currently, we use two correlation methods to examine how the query relates to a theme. Both are based on a standard correlation coefficient. One technique, however, uses the first-order Markov model, while the other simply uses the frequency count of each state observed, which is a zero-order Markov model. Once a comparison has been made to each database entry the results are sorted and presented as a score out of 1000 to the user.

The results so far are promising. We have taken some of the themes extracted automatically and manipulated them to simulate typical errors we expect to see in queries. The three classes of errors that we are investigating are duration change, note drop, and pitch change. In our experiments, we set an error rate for each type of error that indicates what percentage of the MIDI events we will manipulate when comparing the query to the database entries.

[3] We can choose any range of durations, and any method to order the durations.

We define a successful trial as one in which the piece that was being sought is presented as one of the top ten themes returned (a *rank* measure). The system is very robust to duration-change errors, having as high as a 95% success rate even when the error rate is 100%. When it comes to note-drop errors, the system performs at a 75% success rate with error-rates approaching 50%. However the system is not robust to pitch-change errors. It appears that a pitch-change error rate of 10% the system does not return the sought piece in the top rank. We are investigating several solutions to this problem.

We have recently implemented a Viterbi algorithm for retrieval. With this algorithm, we can find which Markov model is most likely to cover the input query. Rather than calculate correlation measures, this approach calculates true posterior probabilities (given the input query as evidence). To account for errors, we insert "error states" in the Markov model for each theme. The distributions for these error states are currently based on our intuition; however, we are conducting an extensive set of experiments to get better error models.

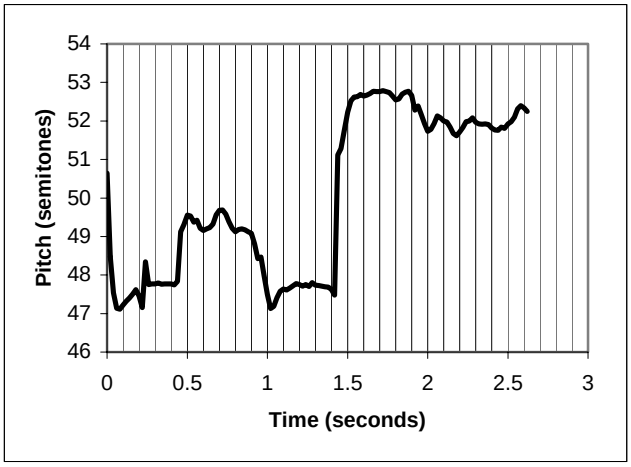
Our initial results with the Viterbi approach are encouraging. Using the same experimental setup as we used for the correlation experiments, we have recorded better results for all error classes. In particular, the Viterbi approach appears to be more robust to pitch-change errors.

The description of both the correlation and Viterbi approaches given in this paper has relied on searching a database of monophonic themes (which do include rhythmic features). Given that both approaches are based on concurrencies, it is very simple to apply both approaches to homophonic or polyphonic music. In fact, we are experimenting with searching for harmonic progressions using the Viterbi approach.

7. FRAME REPRESENTATION AND MELODIC CONTOUR

We are exploring another representation and search strategy to address some of the problems of conventional "event-based" searching, where events are typically musical notes and searching is performed on the basis of note sequences. Event-based searches suffer from at least two problems. First, it is difficult for music transcription systems to segment audio correctly into discrete notes. This is a problematic even when skilled musicians sing queries. Secondly, efficient string-matching algorithms, when given enough leeway to ignore music transcription and performance errors, can often time-align two perceptually dissimilar melodic strings, leading to false positives.

An alternative is to ignore the concept of note and perform a direct comparison of musical contours, representing melody as a function of pitch versus time. This function is discretized by segmenting the melody into time frames. Thus, we call this a "frame-based" approach. Figure 5 illustrates a pitch contour from an audio query and the resulting query string. In this approach, there is no need to quantize pitch, nor is there a problem if pitch varies during a "note" because notes are not represented explicitly. Furthermore, this approach can be extended to incorporate transcription uncertainty by representing pitch as a probability distribution.



Query string: 47.1, 47.4, 47.8, 47.8, 47.8, 49.4, 49.2, 49.6, 49.2, 48.4, 47.4, 47.7, 47.7, 47.7, 51.5, 52.6, …

Figure 5. An audio query is segmented into frames representing a melodic contour. Each frame corresponds to the pitch of a 100 ms timeslice.

To deal with pitch transposition, we transpose queries by many different pitch offsets; to deal with tempo variation, we "stretch" the target melodies in the database by different scale factors. In addition, we use a constrained form of time warping to align queries with the database entries. Constraints prohibit large deviations that would distort the overall contour shape.

This approach is obviously very slow and impractical for searching a large database. However, we believe it would be foolish to limit our research to the existing body of fast search algorithms. Instead, we hope to characterize some limitations of fast search algorithms and then try to overcome them. For example, a slow precise search might be used to refine the results of a fast, imprecise search. To evaluate the frame-based approach, we are replicating various event-based search engines from the literature so that we can compare different approaches using identical queries and databases. Preliminary results show that the frame-based approach is giving substantially better precision. For example, the frame-based approach ranked the correct match as the closest match in 13 out of 20 queries on a collection of 77 big band arrangements, and it ranked the correct match in the top 3 on 16 out of 20 queries. In contrast, an event-based search found the correct match in 5 out of 20 queries, with only 6 out of 20 queries ranked in the top 3 results [19].

8. PHONETIC-STREAM ANALYSIS

In addition to pitch and rhythm, singing as a natural form of query possesses time-varying acoustic-phonetic information, e.g., one sings the *words* of a song according to the pitch and duration of each note. While all three streams may provide useful information for searching the musical database, only the pitch stream has been studied in any detail, and almost no work has been done on the acoustic-phonetic stream. In the ideal case of errorless queries, the acoustic-phonetic stream is likely to be highly redundant with the rhythm and pitch streams, and, therefore, is expected to provide little additional information. In the practical case, where the rhythm and pitch streams may contain a significant number of

errors, the acoustic-phonetic stream may be the only reliable source of information.

In its most general form, extracting the stream of phonetic information from the query is a problem in speaker-independent continuous speech recognition, for which a sizable body of research literature exists, all of which suggests that we should expect little success in the case of sung passages without substantial effort. Besides independence across speakers, the problem of speech recognition for singing is further exacerbated by the fact that non-vocalic segments of the stream are generally poorly represented, e.g., one cannot "sing" the fricative /f/. Furthermore, singing extends the pitch range upwards from the normal range of speaking to fundamental frequencies that generally cause problems for many standard recognition systems.

Our work [20] focuses on a reduced version of phonetic-stream analysis. Rather than attempting to transcribe the word that is sung into standard phonetic units, we have studied coarser quantizations of the phonetic stream, which trade robustness to production variations within and across singer against the information-bearing capacity of the stream. The algorithm we have developed extracts a symbol stream consisting of the Cartesian product of a "phonetic" alphabet of four vowel types (front, neutral, back, non-vocalic) and a duration alphabet of long and short.

Among the several approaches we have studied for segmenting the phonetic stream into the 8-element symbol stream, the most promising appears to be based on a self-referential model, as opposed to an absolute-referential one. In an absolute-referential model, queries are segmented based on templates constructed for the four vowel types over the entire population of singers. A self-referential model segments each query individually and then maps these segments to the most likely "universal" alphabet of front, neutral, back, and non-vocalic. Of the two approaches, the self-referential model appears in our preliminary studies to be more robust to such sources of variation in production as gender, training, song, and register.

The self-referential model utilizes nearest-mean reclassification (NMRA) to segment the query into four categories based on properties of the query's short-time Fourier transform. NMRA is performed on the entire set of short-time Fourier transforms to assign one of four raw categories to each time slice. Aggregation is performed across time slices to yield short and long classification of each vowel type. Finally, the raw categories are mapped into front, neutral, back, and non-vocalic labels based on features of the spectral distributions within and across the raw categories. The string for each query is then compared with the database to find the best matches.

Results from pilot studies suggest that the approach outlined above may be useful in music retrieval. A database of sung queries was constructed by having subjects sing one verse from seven familiar songs: "Happy Birthday", "Yankee Doodle", "America the Beautiful", the Beatles' "Yesterday", "Row, Row, Row Your Boat", "Somewhere Over the Rainbow", and "My Bonnie Lies Over the Ocean". Ten subjects were recruited from among staff, faculty, and graduate students at the Advanced Technologies Laboratory at the University of Michigan. Each subject sang four instances of each song. They were allowed to pace themselves and to choose whatever pitch range and tempo they felt most comfortable for each of the songs. Informal classification of the quality of singing ranged from very poor to excellent across the ten subjects.

Error! Reference source not found. shows the percent correct in the nearest neighbor matches as a function of query song. For each query, the nearest neighbor is found as the vowel stream that requires the fewest number of (weighted) edits to be transformed into the desired query. If that selected vowel stream is generated from the same song as the query, then the selection is deemed correct. There are a total of 40 queries for each song, for a total of 279 possible neighbors from which to select, as we do not count the query itself as a possible neighbor. Therefore, a 14% chance exists of a correct answer occurring at random.

Across all songs, the nearest neighbor selection is correctly chosen from the same song 60% of the time, and varies by song between 42% (for "Yankee Doodle") and 80% (for "America the Beautiful"). We interpret these results as supporting the general hypothesis that some representation of vowel stream is useful for music information retrieval.

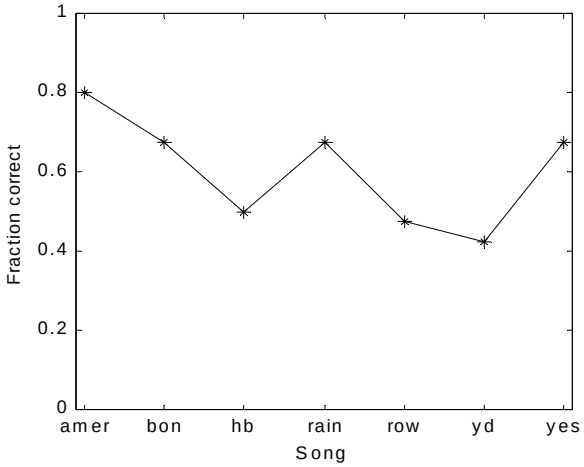


Figure 6. Fraction of correct nearest-neighbor choices as a function of the query song. A correct choice means that the nearest neighbor to the vowel stream is a vowel stream from the same song. There are 40 queries for each song.

9. STRUCTURAL ANALYSIS AND ABSTRACTION

In parallel with our work on melodic search, we are investigating methods by which we can deduce musical structure and genre. Information about structure has many uses. Since repetition is common in music, identifying repetition can aid in music processing and transcription as well as eliminate unnecessary searching. Listeners often remember the most-often repeated parts of a song, so search engines can give extra weight to these, and audio browsers can begin playback at these memorable moments. While a great deal of work has focused on the analysis of symbolic representations of music, we also wish to consider applications to databases of audio waveforms. Both of the approaches described below represent early efforts to derive structure from music audio.

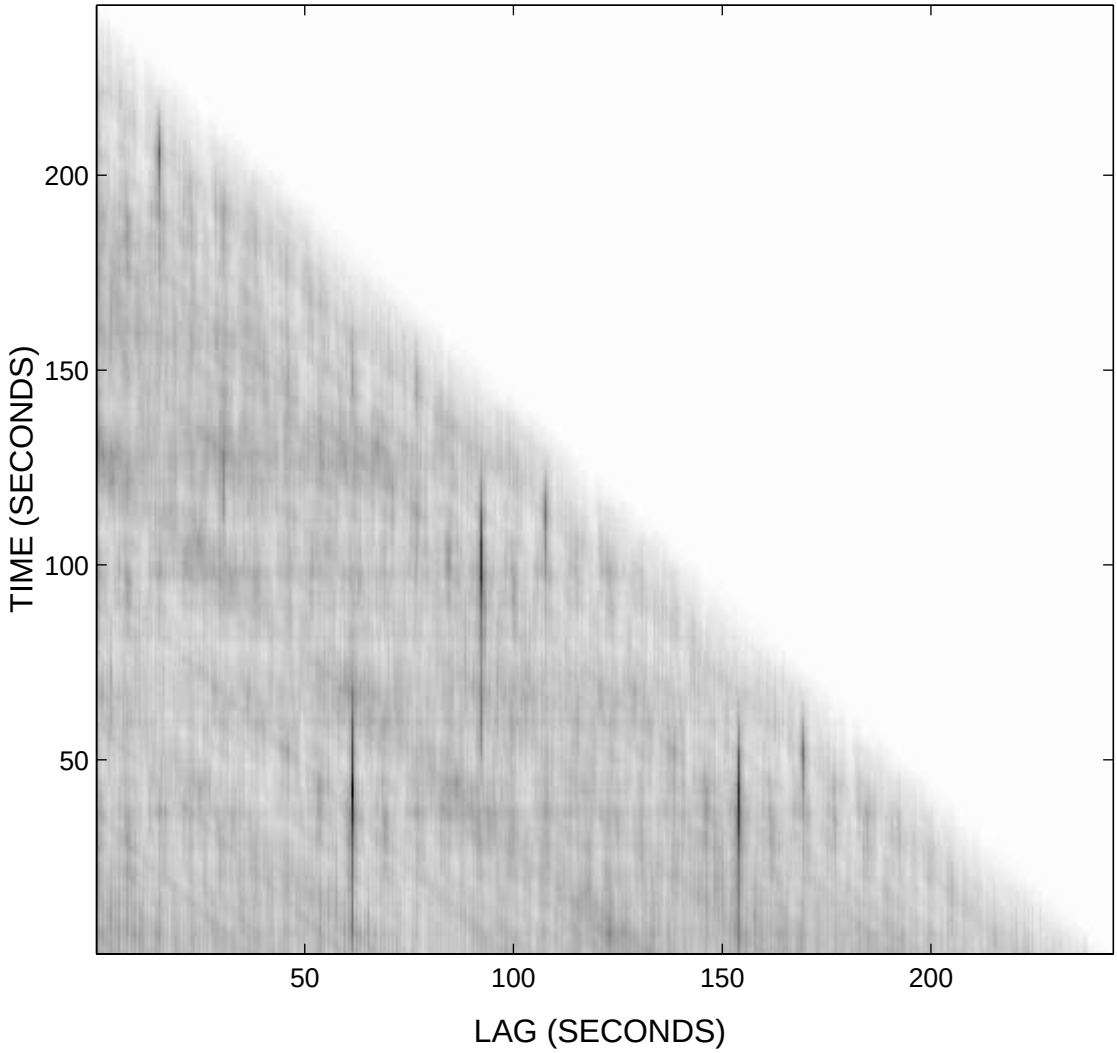


Figure 5. High correlation of chroma at a given lag indicates similarity. Therefore, vertical line segments represent repetition of musical material.

9.1 Chroma-Based Search for Structure

One approach in this direction is our "audio thumbnailing" algorithm [21]. Prior work in this area includes Logan and Chu, [22] who developed algorithms for finding key phrases in selections of popular music. Their work focused on the use of Hidden Markov Models and clustering techniques for mel-frequency cepstral coefficient (MFCC) representations of the acoustic waveform. Their system was subjectively evaluated on a relatively small selection of Beatles songs. In another work, Foote [23, 24] talks about audio "gisting" as an application of his proposed measure of audio novelty. This audio novelty score is based on a similarity matrix, which compares frames of audio based on features extracted from the audio. Foote leaves details such as the similarity metric and feature class as design decisions; however, he does recommends the use of MFCCs as a feature class for computing audio novelty.

Our approach to "audio thumbnailing" draws upon two key concepts: the chromagram and recurrent state. The chromagram is an abstraction of the time-varying spectrum of the audio signal which is based on the perceptual organization of pitch [25]. For each time frame, the chromagram maps the linear-frequency spectrum onto pitch-chroma spectrum, which ranges in semitones over the octave [26]. Among the mathematical properties of the chromagram is that it discounts octave relationships among the components of the frequency spectrum, which are highly redundant in harmonic sources, and places greater emphasis on pitch-class relationships, which bear information about harmony.

Recurrent state abstracts the concept of structural organization in a piece of music. The refrain in a song, for example, is distinguished from the rest of the piece only by virtue of the fact that it, alone, recurs throughout the piece. Thus, in searching for a refrain, or other such structures that repeat often in a piece of music, relatively little can be assumed about the structure save some basic unit of time, which establishes the scale of the structure.

Our system performs audio thumbnailing by examining the audio signal for recurrent states over time scales from 5 to 60 seconds in duration. To reduce redundancies in the representation of

harmonic sources, each time-slice of the chromagram is quantized into twelve equal semitone bins. The 12-dimensional feature vectors are correlated across time and the correlation values are aggregated over windows of time to identify recurrent structures at a particular time scale.

Foote's similarity matrix is a convenient way to represent the feature-correlation space. Windowing, in this case, transforms the similarity matrix into a time-lag surface Figure 7 presents a time-lag surface for Jimmy Buffet's *Margaritaville*. A thumbnail for the piece of music is selected by locating the maximum element of the time-lag matrix subject to two constraints. To prevent the selection of quick repetitions and fading repeats, we require that the location have a lag greater than one-tenth the length of the song and occur less than three-fourths of the way into the song. The thumbnail is then defined by the time position of this maximum, which corresponds to the time that the first of the pair of sections begins, and the length of the window used for aggregating the data.

Our primary quantitative studies of the thumbnailing algorithm have been applied to a database of 93 selections of popular music, with styles including rock, folk, dance, country-western, and others [21]. Each song was hand-scored to identify the refrain or chorus segments of the song. The thumbnailing algorithm was then used to identify highly similar segments of music. Our general observations include the following. With respect to frame-level recall and precision rates, the thumbnail algorithm performs as high as 0.9, for proper choice of window. When it fails, it is often the case that the chorus or refrain is repeated, but there is some change, either in instrumentation or in the musical structure of the repeat. These cases violate our initial assumption of high correlation between instances of the chorus and indicate that this assumption would need to be relaxed under such circumstances. It is also interesting to note that thumbnailing based on the chromagram reduction of the audio stream clearly outperforms a comparable system based on MFCC's. We interpret this outcome to reflect the fact that MFCC's provide a low-order representation of the wideband spectrum, whereas the chromagram provides a low-order representation of the "wideband" harmonic content of the signal, by folding harmonically redundant regions of the spectrum into each other.

9.2 Melodic Pattern Analysis

Another effort in the direction of structural analysis also starts with audio but uses conventional autocorrelation-based pitch estimation to extract melodic contour. The top of Figure 8 shows audio taken directly from a commercial recording of a ballad, "Naima," by John Coltrane and performed by his jazz quartet [27]. Below the audio is a piano-roll display of a pitch transcription, accomplished using a straightforward autocorrelation algorithm for pitch estimation. At the bottom of the figure is the analysis, discussed below.

Taking inspiration from the frame-based melodic contour comparison described in Section 7 and the chroma-based correlation analysis of Section 9.1, the analysis procedure computes the length of similar melodic contours starting at all pairs of locations i and j, which index note position. The matrix $M(i, j)$ is defined as the duration of similar contours starting at locations i and j. ($M(i, j)$ is mostly zero.) For example, there is a repeated 4-bar phrase at the beginning of the piece, starting at the first and seventh notes. Note that where $M(i, j)$ is non-zero, there will tend to be a slightly shorter duration at $M(i+1, j+1)$. For example, there are similar phrases starting at notes 2 and 8. These "implied" entries are "removed" by setting them to zero.

After eliminating these implied entries, clusters of similar melodic contours are formed. For example, similar lengths at i,j, j,k, and k,i imply that three similar contours are located at i, j, and k. In the saxophone solo, there is a third repetition of the opening four bars near end of the excerpt shown in Figure 8.

After simplifying the matrix M and forming clusters from the remaining melodic fragments, a greedy algorithm attempts to "explain" all notes of the transcription in terms of these clusters. The shaded bars at the bottom of Figure 8 locate similar fragments. It can be seen from this that the melody consists of a repeated phrase followed by a shorter repeated phrase and a third phrase. This is followed by a return to the opening phrase. These phrases return after a piano solo (not shown).

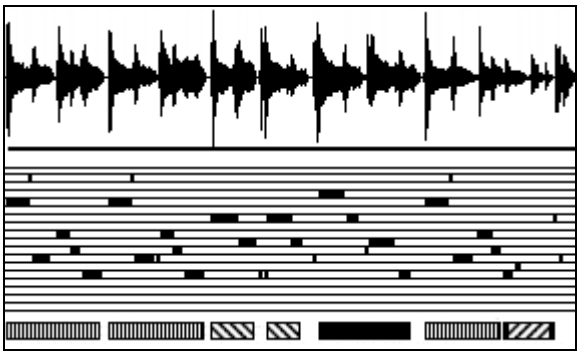


Figure 8. Audio from jazz quartet (top), automatic transcription (middle), and analysis (bottom). Similar shading indicates similar melodic fragments.

This work is at an early stage. It has produced an excellent analysis of a performance of "Naima." This example was chosen because of the clear, simple lines and structure and the dominance of the saxophone in the jazz quartet recording (and also because it is a wonderful ballad). Work is underway to adapt these methods to more challenging examples.

9.3 Genre Classification

Although we have argued that not all searches should be conducted within a given genre, there are certainly many cases where users can narrow their search by specifying music categories. We have investigated the use of machine learning techniques to perform music classification. We trained a neural network classifier on audio power spectra measured within windows of several seconds of duration of different genres. To classify a piece, we divide the piece into many window-sized chunks and run the classifier on each chunk. The overall class is the one reported for the greatest number of chunks. This approach correctly classified all 80 pieces in a database of digital audio as rock, jazz, country, or classical. Much finer classification is desirable, and we hope to incorporate new methods into our architecture as they become available.

10. SUMMARY

We have presented an architecture for music retrieval. The MUSART architecture is motivated by the need to explore and

ultimately rely on multiple mechanisms for representation, abstraction and search. We have made progress toward more robust and flexible music databases. Our work on Markov models provides a new an approach to musical abstraction and retrieval. Our frame-based melodic search and phonetic-stream search deal specifically with problems of audio-based queries. Additional work addresses the problems of audio analysis, the identification of musical structure, and music classification.

In the future, we will refine all of these techniques and integrate them into the architecture we have presented. We also need to explore many user interface issues. The benefits will include the ability to combine multiple search strategies and a more formal approach to the evaluation and comparison of music retrieval techniques.

11. ACKNOWLEDGMENT

This material is supported by NSF Award #0085945 and by an NSF Fellowship awarded to Mark Bartsch. The views expressed here are those of the authors. We thank Jonah Shifrin for implementing tphe Viterbi algorithm and generating the data reported in this paper. We also thank Bryan Pardo and Erica Schattle for comments on previous versions of this paper.

12. REFERENCES

[1] Dunn, J.W., C.A. Mayer. *VARIATIONS: A digital music library system at Indiana University*. in Digital Libraries. 1999: ACM.

[2] Kornstadt, A., *Themefinder: A Web-based Melodic Search Tool*, in *Melodic Similarity Concepts, Procedures, and Applications*, W. Hewlett and E. Selfridge-Field, Editors. 1998, MIT Press: Cambridge.

[3] McNab R.J., L.A.Smith, D. Bainbridge and I. H. Witten, *The New Zealand Digital Library MELody inDEX. D-Lib Magazine*, 1997. May.

[4] Bainbridge, D. *The role of Music IR in the New Zealand Digital Music Library project.* in *Proceedings of the International Symposium on Music Information Retrieval, 2000*.

[5] Tseng, Y.H. *Content-based retrieval for music collections*. in *SIGIR*. 1999: ACM.

[6] McNab R.J., L.A.S., Ian H. Witten, C.L. Henderson, S.J. Cunningham. *Towards the digital music library: tune retrieval from acoustic input*. in *Digital Libraries*. 1996: ACM.

[7] Blum, T., D. Keislar, J. Wheaton, E. Wold, *Audio databases with content-based retrieval*, in *Intelligent multimedia information retrieval*, M.T. Mayberry, Editor. 1997, AAAI Press: Menlo Park.

[8] Rolland, P.Y., G. Raskinis, J.G. Ganascia. *Musical content-based retrieval: an overview of the Melodiscov approach and system*. in *Multimedia*. 1999. Orlando, FL: ACM.

[9] Rothstein, J., *MIDI: A Comprehensive Introduction*. 1992, Madison, WI: A-R Editions.

[10] Hewlett, W. and E. Selfridge-Field, eds. *Melodic Similarity Concepts, Procedures, and Applications*. Computing in Musicology. Vol. 11. 1998, MIT Press: Cambridge.

[11] Sankoff, D. and J.B. Kruskal, eds. *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*. 1983, Addison-Wesley: Reading, MA.

[12] Downie, J. S. *Informetrics and music information retrieval: an informetric examination of a folksong database.* in *Proceedings of the Canadian Association for Information Science, 1999 Annual Conference, 1999. Ottawa, Ontario.*

[13] Zhang, T., C.C.J. Kuo, *Hierarchical system for content-based audio classification and retrieval*, University of Southern California: Los Angeles.

[14] Alamkan, C., W. Birmingham, and M. Simoni, Stochastic Generation of Music, 1999, University of Michigan.

[15] Pardo, B. and W. Birmingham. *Automated Partitioning of Tonal Music*. *FLAIRS, 2000*. Orlando, FL.

[16] Meek, C. and W. Birmingham. *Thematic Extractor*. *International Symposium on Music Information Retrieval* in Second *International Symposium on Music Information Retrieval.* 2001.

[17] Barlow, H. *A Dictionary of Musical Themes.* Crown Publishers. 1983.

[18] Rand, W. and W. Birmingham. *Statistical Analysis in Music Information Retriev*al in Second *International Symposium on Music Information Retrieval.* 2001.

[19] Mazzoni, D., and R. B. Dannenberg, *Melody Matching Directly from Audio* in *Second International Symposium on Music Information Retrieval.* 2001.

[20] Mellody, M., Bartsch, M., and G. H. Wakefield. *Analysis of Vowels in Sung Queries for a Music Information Retrieval System* submitted for publication in *Journal of Intelligen tInformation Systems*. 2001.

[21] Bartsch, M., and G. H. Wakefield. *To Catch a Chorus: Using Chroma-Based Representations For Audio Thumbnailing* in *Proceedings of the Workshop on Applications of Signal Processing to Audio and Acoustics*, 2001.

[22] Logan, B., and S. Chu. Music summarization using key phrases in *International Conference on Acoustics, Speech, and Signal Processing*, 2000.

[23] Foote, J. *Automatic audio segmentation using a measure of audio novelty* in *Proceedings of IEEE International Conference on Multimedia and Expo*, 1999.

[24] Foote, J. *Viusalizing music and audio using self-similarity* in *Proceedings of ACM Multimedia*, 1999.

[25] Shepard, R. *Circularity in judgements of relative pitch* in *Journal of the Acoustical Society of America, 36, 2346-2353*, 1964.

[26] Wakefield, G.H. *Mathematical Representation of Joint Time-Chroma Distributions*. in *Intl. Symp. on Opt. Sci., Eng., and Instr., SPIE'99*. 1999. Denver.

[27] Coltrane, J. Naima on the album *Giant Steps.* Atlantic Records. 1960.