TOWARDS INTELLIGENT STRING MATCHING IN QUERY-BY-HUMMING SYSTEMS

Charles Parker

Oregon State University

ABSTRACT

In the last 5-10 years, there have been several attempts at the creation of a musical database that can be queried acoustically. The main problem hindering such efforts so far has been scalability, with limits of the frequently used three-character representation for melodic contour becoming apparent even at relatively small database sizes. Extending the contour representation to 24 semitone resolution would go a long way towards solving such problems, but applying current string matching methods to a 24 character alphabet would prove difficult at best, and human errors would be far more prevalent. Here we show that the vast majority of errors at this resolution, although more numerous, are quite predictable using current methods of probabilisic modeling. This is evidence which we hope opens the door to more intelligent forms of note matching.

1. INTRODUCTION

Considering the steadily growing size of the literature on Query-by-Humming systems, an accepted case for their possible usefulness seems to already have been made. The commercial applications are obvious, from users trying to pick songs from their personal music collection, to location of a particular song title while in a noisy karaoke bar [1] Another application not often considered is in the realm of copyright law [2], where acoustically matching one song to another already copywritten song before publication could save millions in the cost of litigation. Perhaps the most exciting recent development in the field has been the incorporation of melodic contour information into the new MPEG-7 standard [3].

A survey of papers from the area will reveal a large number of complete implementations [1][4][5][6][7][8] with varying methods and degrees of success. In spite of this proliferation of whole systems, there has been little research directed at the analysis of individual parts of the system. A paper by Downie [9] states that "the literature . . . is one of development, not evaluation." This is in part because the various aspects of a Query-by Humming (QBH) system have their own literature, and are all quite well understood. Pitch extraction, string matching, and database retrieval have all existed for a number of years, and it seems that most of the work involved in building a QBH system is tying it all together.

Unfortunately, the combination of several general algorithms for these problems produces results that are far too general. Most descriptions of QBH system implementations admit that the three character alphabet, which is described in the following section, is not descriptive enough to discriminate over a large number of songs [6][8]. In fact, whole papers [10] have presented evidence dismissing this alphabet as an unscalable method. The remaining question becomes how to solve this problem.

Attempts [1][7] have been made to create a more scalable system by using rhythmic as well as pitch data to query the database. While this approach has shown some promise, is has been shown experimentally [11] that, at least in humans, the predominant factor in song recognition is melody. Even if rhythmic information does prove dependable, such QBH systems using both pitch and rhythm will still benefit from increased melodic resolution. It is in improving this resolution that our work begins.

2. MELODIC REPRESENTATION

Probably the most prevalent method [6][8][2] of melodic representation in QBH systems today is that of the three character alphabet used to describe whether the next pitch in a sequence is of lower frequency, higher frequency, or the same frequency as the current one. For example, the first few notes of "Twinkle, Twinkle" would be represented $RURURDDRDRDRD$ where R means the next note is repeated, D means the next note is at a lower frequency, and U means the next note is at a higher frequency.

In western music, differences in frequency of less than one semitone on the even-tempered scale are ignored. A distance of one semitone on the even-tempered scale is defined as a difference in frequencies f_1 and f_2 where $f_1 > f_2$ and:

$$\frac{f_1 - f_2}{f_1} \approx 0.0561$$

Often, this will be expressed as a function of the logarithms of f_1 and f_2 in order to obtain a meausurement that will grow linearly with the number of semitones. For the purposes of this paper and our discussion of contour however, this definition of semitone will suffice. It will also suffice to use only the western even-tempered scale for this analysis, but an extension to other musical scales is clearly possilble.

There has been much work done in psychoacoustics to demonstrate that melodic contour is the principle factor in the recollection of songs. One of the most oft-sited papers is Dowling's [12]. Readers of this paper trying to build a QBH system must breathe a sigh of relief. Looking for such general changes in pitch is a much less constraining enterprise than is picking out exact intervals, and human errors are so reduced that tracking can often be done to very high degree of accuracy. Indeed, several of the papers listed above saw tracking accuracies in the high 90 percentile range.

Even in Dowling's work, however, he goes on to demonstrate that such a scale is impractical, pointing out scalability as the main obstacle. He sees that on the three-step scale, "Twinkle, Twinkle" above and the Andante from Haydn's "Surprise Symphony" are represented with almost exactly the same characters. He then proposes as a remedy a scale where the number of diatonic steps are indicated along with interval direction. It is also stated that "with familiar melodies, exact interval sizes are precisely remembered", and in fact, that music students often reverse the process - using specific songs to help recall specific intervals.

Clearly, this demonstrates that song recognition is greatly enhanced with the use of a richer alphabet, and that techniques using a three character representation will have problems returning unique results even with small sample sets. The danger of using the full 24-character alphabet, however, lies in the proliferation of errors one might expect with increased resolution, which may even go so far as to make the task of matching the string completely intractable [10]. We will attempt to avoid such dangers by suggesting a form of matching that has probability theory rather than information theory at its core. This assumes, of course, that such errors are not totally (or even mostly) random, which we will determine experimentally.

3. EXPERIMENT IN HUMMING QUERIES

3.1. The Hypothesis

We will attempt to show by experiment that hummed queries will be reasonably accurate in a 24-character alphabet, and that errors made are easily predicted. More specifically, we will show that the vast majority of mistakes made by subjects are mistakes within a semitone of the expected pitch. We also suspect that the presence of large interval jumps will cause a marked increase in errors.

Here we will pause to define a few terms:

A *settling* error is the spurious insertion of a character of type $U1$ or $D1$.

An *off-by-one* error is the substitution of a incorrect character one semitone away from the expected character for the expected character.

A *tracable* error is an error that is either a settling error or an off-by-one error.

The logic behind the hypothesis comes largely from informal listening and intuition, but was formalized in an experiment by Lindsay [13] which found that even untrained singers were able to negotiate large intervals with fair accuracy. It is further supposed that analysis of these errors will yield possible methods of programming QBH systems for more intelligent detection.

3.2. The 24-Character Representation

We will represent melodies to be sung with the full 24 semitone alphabet. Each character has a digit component describing the number of semitones of the jump, and a letter component, U or D describing the direction of the jump. We will eliminate the character "R", which would represent the repeated note. Current methods of detecting repeated notes rely on the subject singing on the syllable "ta" or "da", which guarantee by the nature of the consonant at the outset of each note that a silence of roughly 60 ms will accompany its beginning. Eliminating this constraint will allow subjects to sing on any syllable that is comfortable. This is a relatively common practice among developed systems [5].

To give an example before moving on, the song "Twinkle, Twinkle" from above would be represented as follows.

$$U7U2D2D2D2D1D2D2$$

3.3. Data Collection

We recorded a series of eight subjects, five male and three female. The subjects represented a range of ability, from completely untrained to one who was quite well trained. Pains were taken to assure that the majority of subjects were in the target range of likely users of a finished system, that is, the novice singer with little to no training but a fair sense of pitch. Each one was asked to sing a major scale up and down, the first 12 pitches of "Three Blind Mice", and a series of three interval patterns. These five samples have the following representation in our notation:

$$U2U2U1U2U2U2U1D1D2D2D2D1D2D2 \quad (1)$$

$$D2D2U4D2D2U7D2D1U3D2D1 \quad (2)$$

$$U4U8D8D4U4U8D8D4 \quad (3)$$

$$U6U6D6D6U6U6D6D6 \quad (4)$$

$$U7U5D5D7U7U5D5D7 \quad (5)$$

The subjects were recorded in a quiet environment on a Sony Minidisc Recorder with a high-quality Shure SM57 microphone. The pitches were played on the piano before each utterance to assure familiarity. Digital sampling was done at 44.1 khz and 16-bit resolution. To extract the pitches, we ran a version of Gold and Rabiner's [14] venerable pitch extraction algorithm on successive overlapping frames of the data, tailored to smooth differences of less than a semitone.

4. RESULTS

4.1. Information Theoretical Analysis

We will begin by analyzing our data according the Viterbi algorithm [15] which produces two measures: The correct recognition rate is defined as $(N - S - D)/N$ and the accuracy rate is defined as $(N - D - I - S)/N$ where N is the number of correct characters, D is the number of deletion errors, I is the number of insertion errors, and S is the number of substitution errors.

We first segment the data by processing results in (1) and (2) , which we will call set one, and processing separately for (3), (4), and (5), which we will call set two. Computing the errors in set one yields an accuracy of 80.00% and a correct recognition rate of 86.90%, which is comparable to other recent approaches using similar sample sets [16]. We then run the same analysis against set two. This gives an accuracy rate of -57.14% and a correct recognition rate of 20.41%. Such a drop in accuracy is most certainly supportive of the second part of our hypothesis. We now move to a different type of analysis to consider the errors.

4.2. Bayesian Analysis

We now construct a simple Bayesian network to represent our dataset, shown in figure 1. For a detailed explanation of Bayesian networks, one can refer to the excellent text by Russell and Norvig [17].

For the purposes of this explanation, however, one needs only know that Bayesian networks are a well established method of representing conditional probabilities. For example, we can ask this network the probability that a character is correct given that the character is measured to be a $U6$. We would notate this as follows:

$$P(Error = NoError|Measured = U6) \quad (6)$$

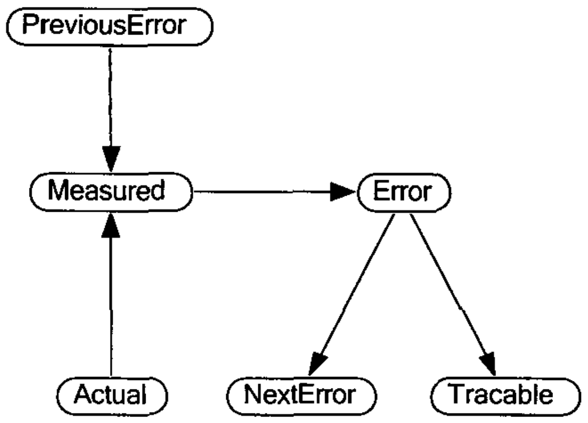

Fig. 1. A Bayesian Network Constructed Over the Collected Data

The three "Error" nodes in the network can take one of the following values:

$$\{NoError, Insertion, Deletion, Substitution\}$$

The "PreviousError" node can also take $Beginning$ as a value if the character is the first in a sequence. Similiarly, the "Tracable" node can take any of the following four values:

$$\{OffByOne, Settling, Other, None\}$$

The two remaining nodes take the values of the 24-character alphabet. The "Measured" node represents the value of the pitch detected, whereas the "Actual" node represents the pitch we *expected* to detect at the same character. If they match, "Error" = $NoError$ and "Tracable" = $None$. If they are one semitone apart, "Error" = $Substitution$ and "Tracable" = $OffByOne$.

Commercially available Bayesian network tools such as Netica [18] allow us to input a dataset, *set evidence* on any of these nodes, and then observe another node to note how probabilities change. For example, to query the network about the value in (6), we simply set the "Measured" node to $U6$ and observe the probability of the outcome $NoError$ on the "Error" node.

We do the analysis twice, first on set one alone, and second on sets one and two combined. The following conditional probabilities are measured: The probability of the error being tracable given that the note is an insertion error, the probability that the note is an insertion error given that it is a U1 or D1, and the probability an error is tracable given that the note is a substitution error. For reference, we also include the probability that *any* note is an insertion error given no other evidence. The results are summarized in table 1 and are conclusive with regards to the hypothesis.

We see that roughly 80% of substitution errors and 85% of insertion errors appear to be of the tracable types. We also see that insertion errors, as expected, are far more prevalent in the combined analysis (with the presence of large intervals) than in the analysis of set one alone. Another interesting note is that *all* of the insertion errors in set one were within one semitone. This suggests that not only do smaller interval sizes promote accuracy, but they also shorten the distance of the errors made. The statistic showing that, over both datasets, simply measuring a character to be $U1$ indicates a better than 75% chance that it will be an insertion error is a bit of a Trojan horse, as there were no $U1$ or $D1$ characters expected in set two, and so all of them will naturally be erroneous. In set one however, there were five such characters expected for each subject, and so the measurement is a more legitimate one.

4.3. A Simple Application

Armed with the knowledge gained from the model, we return to set one, which is most like the queries which will ostensibly be fed into a QBH system (in terms of the nature of the queries with regard to interval size). Since it appears that the $U1$ and $D1$ characters form a disproportionately large number of errors in the string, we will simply post-process the query strings and delete all occurances of $U1$ and $D1$, doing the same to the two strings against which we are matching, producing the following:

$$U2U2U2U2U2D2D2D2D2D2$$
$$D2D2U4D2D2U7D2U3D2$$

We recompute the accuracy and correct recognition rates and obtain 87.27% for both measures. Thus, even by using this data in a very primative fashion, we see an appreciable increase in accuracy. Perhaps even more notable is the fact that all insertion errors have been eliminated. We wish to stress at this point, however, that the true power of this model does not lie in this application, but in an application of a more advanced variety as described below.

5. CONCLUSIONS AND FUTURE WORK

This research seems to indicate that generally, a semitone resolution is too much to demand of a human singer. Certainly, the evidence supports this claim, but one could also read the research as indicating that a three, five or even seven character resolution is far too *little* to demand of a human singer. However, since the $U1$ and $D1$ characters are so close together, and so close to the boundry between up and down, it seems that settling errors could wreak havoc on alphabets of any size. By considering all 24 characters within a two octave range, however, we can "see the whole picture" so to speak, get rid of spurious insertions, and reduce to a coarser alphabet if need be.

But this is again ignoring the primary thrust of the research. It is the Bayesian analysis itself that can be applied to any candidate algorithm to identify weaknesses and regularity in errors. The simple table in this paper barely scraches the surface of interesting relationships between interval and correctness the model helped us to discover. One of the most remarkable was that, given a substitution error, there was a better than 50% chance the next note would be an insertion error, corresponding to the case where the singer jumps to a new pitch, quickly realizes his error, and corrects by moving up or down. Another was that the overall probability of an incorrect value jumped more than 20% on the first note of the utterance, which one might expect just as a subject begins to sing. Thirdly, although the evidence has shown that the original hypothesis was correct, we did not consider extending the hypothesis until we viewed the processed data. Tracing all errors within 2 semitones (i.e., substitutions of +2 or -2 and insertions of $U2$ or $D2$ as well as the one-semitone errors) gives us a tracability of up over 95% for both measures. Increasing this yet again to three semitones yields 100% tracability: There were no substitution or insertion errors of more than three semitones. With the information that, given a character, there are only 6 possible substitutions

Probability Query	Set One Results	Combined Results
$P(Tracable = Settling \mid Error = Insertion)$	1.00	0.845
$P(Tracable = OffByOne \mid Error = Substitution)$	0.882	0.800
$P(Error = Insertion \mid Measured = U1)$	0.364	0.754
$P(Error = Insertion \mid Measured = D1)$	0.097	0.509
$P(Error = Insertion)$	0.048	0.169

Table 1. Results Obtained from Queries to the Network of Figure 1

out of 24 that have a reasonable chance of occuring, we begin to see the usefulness of this techinique.

Many of the current methods for database matching with the three character alphabet are standard dynamic programming approaches such as *n-gram*, *longest common substring*, and *longest common subsequence* (The reader is encouraged to consult [2] for a brief but solid discussion of each). While such techniques are feasible for a small alphabet, they are not efficient enough to handle the large alphabet. In the absence of a large alphabet matching technique, we can look to very successful probabilistic models being used in speech recognition and production as possible inspiration. Given the success that our model has shown in producing discernable trends in the production of errors, we are certain that any comprehensive system for contour recognition would do equally well to utilize a probilistic model.

Of course, for the model to be comprehensive, it must be augmented and tested further. Certainly the largest weakness of the model is the small amount of data, and more must be obtained. We must also explore the possibility of other probabilistic dependancies. For example, preliminary data indicates that the length of the pitch to be sung drastically effects the probability of its being sung correctly. In addition, it does not escape us that the optimal model would probably have both "Actual" and "Measured" connected to the error node, producing a full table of the likehood of any measured pitch given the expected pitch. We also must obtain prior probabilites over note values, a task which has already been accomplished in [1] and [7]. Efforts to refine the model and actually employ it for string matching are currently underway.

In summary, we have shown quite conclusively that errors in humming queries are highly regular, highly dependent on interval size, and that error trends can be accurately detected with a Bayesian network. We have also made a case for the use of a Bayesian network as part of the matching aspect of a QBH system. Given these observable trends in the nature of produced errors, it is clear that higher resolutions of melodic representation are possible in QBH systems, but only if more intelligent forms of error detection and correction such as this are employed.

6. REFERENCES

[1] Naoko Kosugi et. al., "A practical query-by-humming system for a large music database," in *Proc. 8th ACM Multimedia Conference*, 2000.

[2] Alexandra Uitdenbogerd and Justin Zobel, "Matching techniques for large music databases," in *Proc. 7th ACM Multimedia Conference*, 1999.

[3] MPEG Requirements Group, "Overview of the mpeg-7 standard," in *Doc. ISO/MPEG N4031, MPEG Singapore Meeting*, March 2001.

[4] Dominic Mazzoni and Roger B. Dannenberg, "Melody matching directly from audio," in *Proc. 2nd Annual International Symposium on Music Information Retrieval*, 2001.

[5] Lei Lu et. al., "A new approach to query by humming in music retrivial," in *Proc. IEEE International Conference on Multimedia and Expo*, August 2001.

[6] Preeti Rao and M. Anand Raju, "Building a melody retrieval system," in *National Conference on Communications*, Bombay, India, 2002.

[7] Youngmoo E. Kim et. al., "Analysis of a countour-based representation for melody," in *Proc. 1st International Symposium on Music Information Retrieval*, October 2000.

[8] Asif Ghias et. al., "Query by humming: Music information retrieval in an audio database," in *Proc. 3rd ACM Multimedia Conference*, 1995, pp. 231–236.

[9] Stephen Downie and Prof. Michael Nelson, "Evaluation of a simple and effective music information retrival method," in *Proc. 23rd International ACM SIGIR conference on Research and Development in Information Retrieval*, 2000.

[10] Timo Sorsa and Jyri Huopaniemi, "Melodic resolution in music retrieval," in *Proc. 2nd Annual International Symposium on Music Information Retrieval*, 2001.

[11] D. J. Levitin, *Memory for Musical Attributes*, pp. 214–215, MIT Press, Cambridge, MA, 1999.

[12] W. Jay Dowling, "Scale and contour: Two components of a theory of memory for melodies," in *Psychological Review*, 1978, vol. 85, pp. 341–354.

[13] A. T. Lindsay, "Using contour as a mid-level representation of melody," M.S. thesis, MIT Media Lab, 1996, Unpublished.

[14] B. Gold and L. Rabiner, "Parallel processing techniques for estimating pitch periods of speech in the time domain," *Journal of the Acoustical Society of America*, vol. 46, pp. 442–448, 1969.

[15] A. J. Viterbi, "Error bounds for convolutional codes and an asympotitcally optimum decoding algorithm," in *IEEE Transaction on Information Theory*, 1967, vol. IT-13, pp. 260–267.

[16] Hsuan-Huei Shih et. al., "An hmm-based approach to humming transcription," in *Proc. IEEE International Conference on Multimedia and Expo*, 2002.

[17] Stuart Russell and Peter Norvig, Eds., *Artificial Intelligence: A Modern Approach*, Prentice Hall, 1995.

[18] Norsys Software Corp, "Netica: Proabilistic modeling software," On the World Wide Web at www.norsys.com, 2002.