音 楽 情 報 科 学 49−15
（２００３．２．２２）

複数の音高候補値を用いた楽曲検索システムの構築

許 盛弼† 鈴木 基之‡ 伊藤 彰則‡ 牧野 正三‡

†東北大学 大学院情報科学研究科，‡東北大学 大学院工学研究科
〒980-8579 仙台市青葉区荒巻字青葉 05
E-mail: †hsphil@makino.ecei.tohoku.ac.jp, ‡{moto, aito, makino} @makino.ecei.tohoku.ac.jp

あらまし 本報告では，ハミング入力による楽曲検索システムの構築を行ったので報告する．システムでは音長や音高を特徴量として，ハミングと楽曲とのマッチングを連続 DP によって行う．このうち音高については，従来の方法ではピッチ抽出を行うことで１つの音高を決めていたが，本システムではピッチ抽出の誤りを考慮してピッチ抽出の際の複数の音高候補値と信頼度を用いる．複数の音高候補値をことで，用いない場合と比べて検索性能が向上した．また，実験で同様のシステムとの検索性能の比較を行った結果，提案するシステムは同様の楽曲検索システムと比べよい検索性能を示した．

キーワード 楽曲検索，ハミング，複数音高値候補

Construction of the Music Retrieval System using the Multiple Pitch Candidates

Sungphil HEO†, Motoyuki SUZUKI‡, Akinori ITO‡, and Shozo MAKINO‡

† Graduate School of Information Sciences Tohoku University, ‡ Graduate School of Engineering Tohoku University
05 Aza-Aoba, Aramaki, Aoba-ku, Sendai, 980-8579 Japan
E-mail: †hsphil@makino.ecei.tohoku.ac.jp, ‡{moto, aito, makino} @makino.ecei.tohoku.ac.jp

Abstract Users do not sing accurately, especially if they are inexperienced or an accompanied; even skilled musicians have difficultly in maintaining the correct pitch of a song. Moreover errors may occur when a musical retrieval system extracts pitch from humming. Consider of these problems, we propose to extract multiple pitch candidates. This method has shown that multiple pitch candidates are important features in determining melodic similarity, but it is also clear that reliability information which obtained from power is important as well.

In the experiment, we compared to search efficiency of the similar system. Proposed method showed good retrieval result compared with the similar system.

Keyword Music Retrieval System, Humming, Multiple Pitch Candidates

1. まえがき

コンピュータハードウェア及びマルチメディア情報処理技術の発達によって，マルチメディア情報検索システムに対する要求が徐々に増加してきている．このようなマルチメディア情報の検索方法はテキストやイメージを用いたものを主として発達して来たが，動画とオーディオ情報，特に音楽情報に対するマルチメディア情報検索システムに対する要求が増加してきている．

最近の音楽情報検索システムの研究では，該当の音楽情報の歌手名，作曲家，曲のタイトルなどのテキスト情報のみを利用した検索だけでなく，音楽情報を利用した検索に対する研究が成されている[1,2,10]．しかし，ユーザの利便性や検出精度の面ではまだ不十分である．過去の研究では，音楽情報として音の長さ（音長）や音の高さ（音高）といった情報が用いられている．このうち音高はピッチ抽出によって得られるが，一般にピッチ抽出自体の精度が十分でないことが検索性能の上がらない原因のひとつとして考えられる．この問題を解決するために，ピッチ抽出の誤りを考慮して，複数の音高候補値を用いる楽曲検索システムを提案する．

本システムではユーザの入力したハミングから，旋律を構成する要素である音長，複数の音高値候補，信頼度を複合的に用い，データベース中の楽曲とハミングとのマッチングを連続 DP を用いて行う．

ユーザのハミングにはキーの違い，テンポの違いなど個人性による影響や，音符の挿入・脱落のような歌唱誤りなどが含まれるため，マッチングの距離尺度やDP のパスを工夫することで,これらの問題の解決を図った．

また，提案方法の評価のために同様の楽曲検索システムの方式[2]を実現して検索精度の比較を行なった．

2. 複数の音高値候補を用いた楽曲検索システムの概要

2.1. 楽曲検索システムの構成

本システムはユーザの歌唱から抽出された旋律の音高列と音長列をデータベースとマッチングすることにより検索を行なう．

曲検索システムの概要を図 1 に示す．ユーザはまず，マイクを使用してメロディをハミングし，楽曲情報の入力を行う．システムでは入力されたハミングから 1 音符の区間を検出した後，音高・音長などの特徴量を抽出し，その特徴量を用いてマッチングを行い，入力したハミングと最も距離が近い楽曲を検索結果とする．

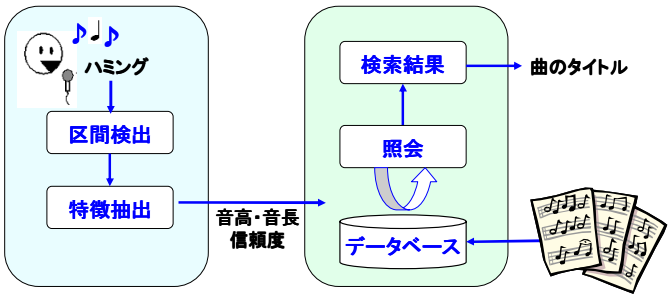

図 1：楽曲検索システムの構成

2.2. 特徴量の抽出法

2.2.1. 音符の区間検出法

最も簡単な音符単位の区間検出の方法としては入力歌唱に対して平均振幅を用いた適当な閾値処理を行なうものがある．その時実際 2，3 個の音符の区間が 1 個の区間にセグメンテーションされたり，また 1 音符の区間が 2,3 音符区間に判定されたりする可能性がある．その結果，音長と音高の抽出誤りが発生し検索精度の大きな低下の原因になる．

図 2 にハミングの波形とセグメンテーションの例を示す．図 2 では,実際には 2 音か 3 音が検出されなくてはならないところが 1 音として検出されたり（A, C）,1 音符の区間が 2 音符区間に判定されたり（B）場合が発生することを示している.音長値と音高値の抽出をできる限り忠実に行うためには，1 音符のセグメンテーションを正確に行う必要がある．

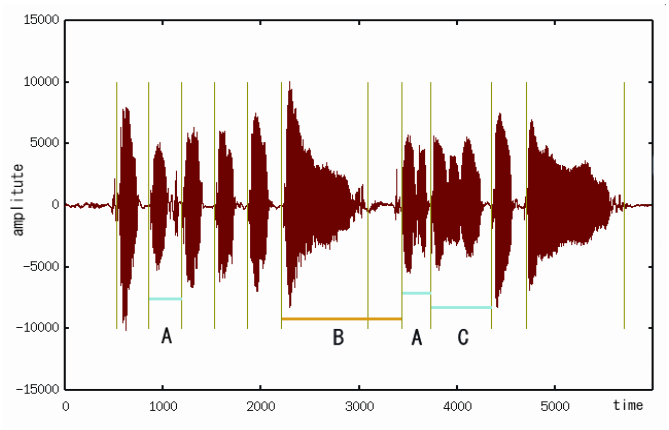

図 2：区間検出誤りの例

本論文では，

（1）歌唱法を/ta/に限る

（2）パワーの差分に注目する

ことで，高精度な区間検出を行なう．

歌唱法を/ta/に限定し，差分フィルタを掛けることで，無声音かつ破裂音/t/と有声音/a/の間でのパワーの時間的変動のエッジ抽出を行った．

歌唱法を/ta/に限定することで自由に歌うことかできないためユーザが不便に感じることになるが，この歌唱方法は無理がないためユーザにとって特に負担とはならない[2].

セグメンテーションにおいて，歌い出しの部分でマイクに吹きかかる呼吸音を 1 音符として誤って抽出することがある．これに対しては，/a/を検出するのに特化し/a/のフォルマント（600～1,500Hz） に対応する帯域通過フィルタをかけることで，/a/の部分を強調し，/a/では無い部分を誤って検出することを防ぐこととした．区間検出の処理の流れとしては，図 3 のようになる．

図 3 ：フィルタ処理による区間検索の方法

2.2.2. 音長情報の抽出

音符の区間を検出することで音長情報を抽出することができる．この時，音符の始まりから音符の終了までを音長とする方法（オン-オン）と，音符の始まりから次の音符の始まりまでを音長をする方法（オン-オフ）が考えられる．

実際に 4 分音符をハミングした場合，ユーザによって 4 分音符の長さでハミングする人もいるが，短くハ

ミングをして 4 分音符の長さだけ休んだ後，その次の音符をハミングする人もいると考えられる．つまり，同じ曲でもユーザによってハミングのスタイル(スタッカート形態のハミング)やハミング速度の違い(速く，遅く)がある．

そこで，オン-オンとオン-オフで，どちらのほうがよりユーザのハミングスタイルに対応しているかを調べるために次のような方法でハミングデータの分析を行った．

5 名のユーザがハミングした曲に対してハミングと対応する MIDI の音符の音長の関係を分析した．

ハミングのテンポを MIDI とそろえるため，ハミングした音符の平均音長が対応する MIDI 区間の平均音長に等しくなるように式(１)を用いて正規化をした．ここで，*HUM(i)*は *i* 番目のハミングの音符の長さ，*MIDI(i)*は *i* 番目の MIDI の音符の長さを表す．式（２）の $HUM_s(last\text{-}note)$と $HUM_s(first\text{-}note)$はそれぞれ，ハミングの最後の音符の始まり時刻と最初の音符の始まり時刻し，$MIDI_s(last\text{-}note)$と $MIDI_s(first\text{-}note)$はそれぞれ，ハミングされた MIDI の最後の音符の始まり時刻と最初の音符の始まり時刻である．

正規化された音長が MIDI の音長に対してとの程度の長さを持つかという相対音長を式（３）によって計算した．相対音長の分布を図４に示す．

$$norm\_HUM(i) = HUM(i) * norm\_ratio \quad (1)$$

$$norm\_ratio = \frac{MIDI_s(last-note) - MIDI_s(first-note)}{HUM_s(last-note) - HUM_s(first-note)} \quad (2)$$

$$relative\_length(i) = \frac{norm\_HUM(i)}{MIDI(i)} * 100 \quad (3)$$

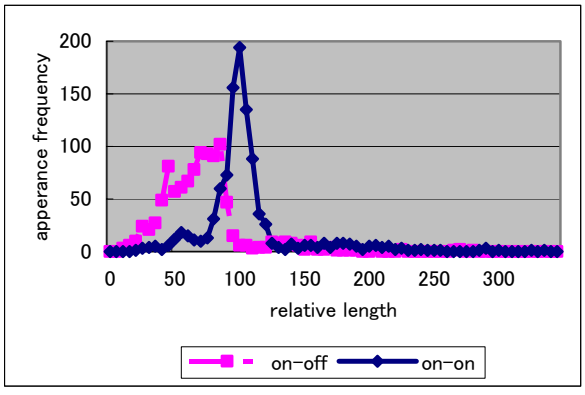

図４：ハミングオン-オフとオン-オンに対する音符の分布

オン-オフの場合には相対値 100 に対する出現頻度が少なく，また分散が広がっているので実際 MIDI の長さと違う傾向が多い．

これに対して，オン-オンの場合には相対値が 100 のところでピークがあり，ユーザの癖を吸収するのに適合したパラメタであるといえる．そこで，音長の場合にはオン-オンを特徴量として使用する．

2.2.3. 音高情報の抽出

音高の特徴量には各音符の音長区間の中で最も安定している中心フレームから図５のようにピッチ抽出[11]を行い，得られた複数の音高値の候補を用いる．

各音高値候補それぞれのパワーを信頼度の高い第 1 位候補の値を用いて正規化し，これを信頼度として距離尺度に導入する．

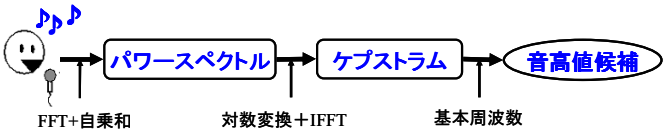

図５：音高値候補の抽出方法

2.2.4. 音高・音長情報の相対化

ユーザによって入力された歌声は調やテンポがデータベースと異なることが考えられるため,なんかの正規化を行う必要がある．ここでは直前の音符との相対音高比・相対音長比を計算することで正規化を行なう．この時，音符の区間検出の誤り等によって音符の挿入や脱落が起こった場合，比を取る相手か変わるために特徴量が大きく変化してしまうという問題がある．

音高については，例えば「ド・ミ」という音を「ド・ミ・ミ」と誤って音符の挿入が起きた場合，相対音高比は「$^{ミ}/_{ド}$」ではなく「$^{ミ}/_{ミ}$」となってしまう．

そこで，実際にデータベースとマッチングする際，音符の挿入や脱落が起きた場合はそれに従って比を取る相手を動的に変化させる（この例の場合は２つ前の「ド」との比を取る）ことで正しい相対音高比を抽出する．

一方，音長の場合は比を取る相手を変化させるだけでは問題の解決にはならない．

データベースの１音符を２つ分けてハミングした場合とデータベースの２音符の区間を一つにつなげてハミングした場合の例を図６に示す．

データベースの「m3」「m4」をひとつの音符としてハミングした場合(HUM sequence A)，ハミングの「a3」の音符の相対音長比は「$^{a3}/_{a2}$」と対応するのはデータベースの「m4/m2」ではなく「$^{m3+m4}/_{m2}$」となる必要がある．同様にデータベースの「m3」を２つの音符としてハミングした場合(HUM sequence B)も，データベースの「$^{m3}/_{m2}$」に対応するのは「$^{b3'+b3''}/_{b2}$」である必要がある．

そこで，相対音長比を取る場合は挿入や脱落に従って対応する音長の和をとり，それを相対化することで，正しい相対音長比を得る．

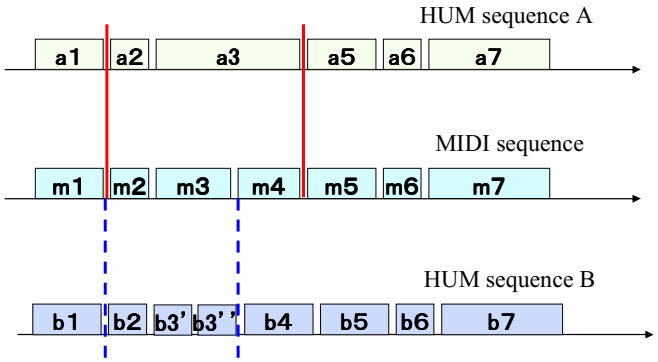

図 6 ：ハミングの誤りの例

2.3. 検索アルゴリズム

ハミングから得られた特徴量とデータベース中の楽曲の特徴量を連続 DP を用いてマッチングを行い，DP スコアが最小の楽曲を検索結果とする．

マッチングアルゴリズムには，

（1）音符の挿入や脱落がある

（2）データベース中のどこから歌い出すか不明である

ことから連続 DP を用いた．

連続 DP の式を式（4）に示す．式（4）で $g(i,j[k])$ は楽曲データの i 番目の音符とハミングの j 番目の区間での k 番目の音高候補値の累積距離である．また，式（5）の $d_z(z = 1,2,3)$はそれぞれ図 7 の DP パスに対応する．また，p_z，c_z，t_z は各 DP パスでの音高値間の比，正規化信頼度，音長値間の比である．

$$g(i,j[k]) = \min \begin{cases} \min_l \{g(i-2, j-1[l]) + d_1(i, j[k,l]\} \\ \min_l \{g(i-1, j-1[l]) + d_2(i, j[k,l]\} \\ \min_l \{g(i-1, j-2[l]) + 2 * d_3(i, j[k,l]\} \end{cases} \quad (4)$$

$$d_z(i,j[k,l]) = \beta\{\alpha p_z(i,j[k,l]) + (1-\alpha)c_z(j[k,l])\} + (1-\beta)t_z(i,j) \quad (5)$$

ここで，p_z，c_z，t_z はそれぞれ以下のように定義される．

$$p_1(i,j[k,l]) = \left| \log\left\{\frac{mid_d(i)}{mid_d(i-2)}\right\} - \log\left\{\frac{hum_d(j[k])}{hum_d(j-1[l])}\right\} \right|$$
$$p_2(i,j[k,l]) = \left| \log\left\{\frac{mid_d(i)}{mid_d(i-1)}\right\} - \log\left\{\frac{hum_d(j[k])}{hum_d(j-1[l])}\right\} \right| \quad (6)$$
$$p_3(i,j[k,l]) = \left| \log\left\{\frac{mid_d(i)}{mid_d(i-1)}\right\} - \log\left\{\frac{hum_d(j[k])}{hum_d(j-2[l])}\right\} \right|$$

$$c_1(j[k,l]) = hum_p(j[k]) + hum_p(j-1[l])$$
$$c_2(j[k,l]) = hum_p(j[k]) + hum_p(j-1[l]) \quad (7)$$
$$c_3(j[k,l]) = hum_p(j[k]) + hum_p(j-2[l])$$

$$t_1(i,j) = \left| \log\left\{\frac{mid_t(i-1) + mid_t(i)}{mid_t(i-2)}\right\} - \log\left\{\frac{hum_t(j)}{hum_t(j-1)}\right\} \right|$$
$$t_2(i,j) = \left| \log\left\{\frac{mid_t(i)}{mid_t(i-1)}\right\} - \log\left\{\frac{hum_t(j)}{hum_t(j-1)}\right\} \right| \quad (8)$$
$$t_3(i,j) = \left| \log\left\{\frac{mid_t(i)}{mid_t(i-1)}\right\} - \log\left\{\frac{hum_t(j-1) + hum_t(j)}{hum_t(j-2)}\right\} \right|$$

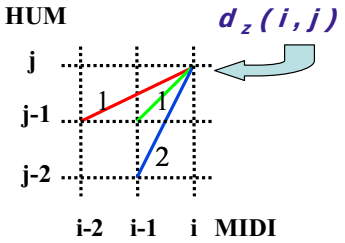

図 7 ：DP パスと重み値

通常の DP マッチングの場合にはある点 $d_z(i, j)$での値は i, j のみに依存し，DP パスによって変化することはない．しかし，式（5）は DP パスによって同一点 $d_z(i, j) = d_z$ に対する値が変わる．これは DP パスの違いによって対応する音符がかわるため相対音高比や相対音長比を動的に計算しているためである．

3. 評価実験

実験では検索キーの組み合わせによる性能評価を行なった．特に，音の高さの特徴量として複数の音高候補値を使うことで検索性能がどの程度変わるかを調べるための実験を行った．また，重み係数(α，β)を変更しながら楽曲検出率に対する実験を行った．

3.1. 実験条件

音楽データベースとしては，MIDI データから得られた童謡 155 曲のデータベースを用意し，男性 5 人がハミングしたものを使って実験を行なった．実験条件を表 1 に示す．

表 1 ：実験条件

楽曲データベース	童謡 155 曲（MIDI）
テストデータ	男性 5 名による歌唱（320 曲）
サンプリング周波数	16kHz
分析窓	64 ms ハミング窓
分析周期	8ms
区間検出	帯域通過フィルタ ：600～1，500Hz 差分フィルタ ：1 次微分
特徴量	音高値候補，信頼度，音長値

3.2. 音符の区間検出精度の評価

提案した区間検出の方法で，セグメンテーション誤りがどの程度かを調べた．セグメンテーション誤り率を式（9）で定義する．320 曲中の 20 曲の歌唱に対する音符区間の検出の結果，セグメンテーション誤り率の平均は 3.5%であった．ほとんどのユーザはセグメン

テーション誤りがわずかであったが，歌唱者 4 だけが他のユーザと比べてセグメンテーション誤りが多かった，ほかのユーザについてはほとんど良い結果を得た．

歌唱者 4 のセクメンテーションの誤りの大きな原因としてはハミング時/ta/ではなく別の歌声でハミングした部分があり，その部分に誤りが多かった．

$$\text{セグメンテーション誤り率（\%）}=\frac{\text{挿入数}+\text{脱落数}}{\text{正解音符数}}\times 100 \quad (9)$$

表２：セグメンテーション誤り率

ユーザ	正解音符数	挿入数	脱落数	誤り率(%)
歌唱者 1	514	4	1	1.0
歌唱者 2	495	9	2	2.2
歌唱者 3	431	10	2	2.8
歌唱者 4	438	24	15	8.9
歌唱者 5	433	9	5	3.2
合計	2311	56	25	3.5

3.3. ハミング中の音符数の分布

今回の実験では被験者に曲のタイトルのみ提示し，自由に歌ってもらったため，音符の数にバラつきが見られた．

図８はユーザがハミングした音符数の分布である．一番短く歌った曲は 4 音符（曲名：あいあい）から一番長く歌った曲の音符数は 30 音符（曲名：まっかなあき）であった．平均的には 11.6 音符程度ハミングをした．

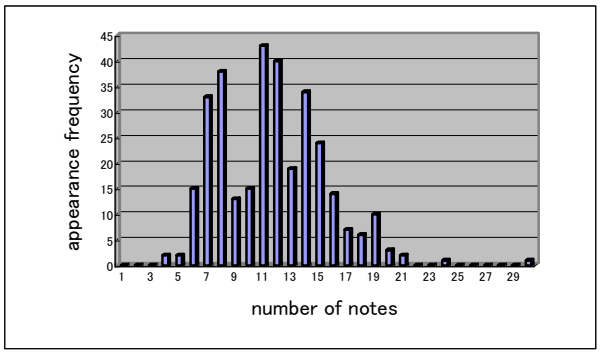

図８：ハミングした音符数の分布

3.4. 楽曲の検索実験

3.4.1. 評価方法

評価には正答率を用いた．正答率は，与えられた順位以内に同じ距離値を持った複数の曲が存在し，検索結果としてより多くの曲を出力した場合は，同距離の候補曲から乱数を用いて選択する，とした時の正解率として定義する．例えば，検索した結果，1 位になった曲が n 曲あった場合は $1/n$ として計算した．

$$\text{正答率(\%)}=\frac{\text{同一順位についてランダムで選択}}{\text{総ハミングした曲数}} \quad (10)$$

3.4.2. 検索精度の評価

実験では特徴パラメタの種類を変更しながら楽曲検索率に対する実験を行った．重み係数(α，β)を変更しながら実験を行った結果，最もよい結果のみを表３に示す．表３中の音高（m）は音高候補値を m 個用いたことを示す．また，比較として文献[２]の方法を用いて同様の実験を行った結果も合わせて示す．文献[２]では検索精度を上げるために Coarse-to-Fine 法を用いているが，ここでは Coarse-to-Fine 法による結果(A)と，最初から 27 段階のカテゴリーに分けた場合(B)を合わせて示す．

特徴量として音長・音高のみを利用したより多様な特徴量を共に利用した場合より良い結果を得た．

表３：特徴量と正答率（%）

特徴量	1 位	5 位以内	10 位以内	重み
音長のみ	42.6	65.2	71.2	β =0.0
音高のみ	66.1	83.6	90.4	α =1.0 β =1.0
音長＋音高	73.9	87.5	91.1	α =1.0 β =0.6
音長＋音高(1)＋信頼度	81.3	89.6	91.8	α =1.0 β =0.6
音長＋音高(3)＋信頼度	86.5	90.3	94.1	α =0.5 β =0.7
音長＋音高(5)＋信頼度	83.3	91.2	93.2	α =0.6 β =0.7
音長＋音高(7)＋信頼度	78.6	92.7	95.5	α =0.6 β =0.8
A:Coarse-to-Fine	78.4	83.1	89.6	無
B:Category 27	81.6	88.1	91.5	無

音高候補値が 1 の場合，信頼度を入れることで，特に 1 位の正答率が向上しており，その有効性が確認できた．音高候補値を 3 個用いた場合に 1 位正答率で最も良い結果である 86.5%が得られた．また，音高候補値を 7 個用いた場合に 10 位以内正答率で最も良い結果である 95.5%が得られた．

音高候補値を複数用いることで，文献[2]で最も性能の良かった方法（B）と比べてもより高い正答率が得られた．また，音高値が 1 つである場合は（B）と同程度の性能であることから音高候補値を複数用いることで性能が向上していることがわかった．

3.5. 検索時間

候補点を増やすことによって検索精度が向上になるが検索時間に影響を及ぼすことになるのでこれに対する分析をした．

Alpha21264 600MHz を用いて計算された時の一曲に対する平均検索の時間を図９に示す．抽出音高値候補点数が多くなると指数関数的に時間が増加した．ハミングした音符数が多いものは時間がかかっていた．今後，更にデータベースの曲数を増やす場合は検索時間が多くかかることか予想される．そこで高速化法が必要であると思われる．

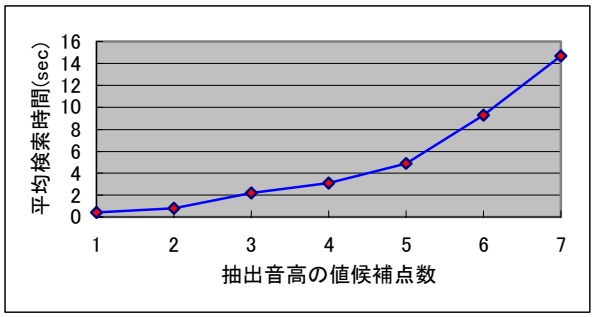

図９：音高値候補点数と平均検索時間の関係

4. むすび

旋律を構成する要素である音長，多数の音高候補値，信頼度情報を抽出して，これらを相互結合してデータベースの間の類似度を計算することで高精度に楽曲を検索するアルゴリズムを提案した．

音長のみや音高のみを利用した場合より音長と音高だけではなくてパワー成分も共に利用することで高精度な検索が行なわれた．

特に，音高候補値を１つだけ用いる場合と比べて，複数の音高候補値を用いることでより高い正答率が得られた．また，同様の楽曲検索システムである文献[２]の方法に比較してもより高い正答率が得られた．

今度は提案された特徴量とマッチングアルゴリズムの検証の為に多様なジャンルの音楽を対象に実験を行う予定である．

文　献

[1] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, “Query By Humming - Musical Information Retrieval in an Audio Database”, Proc. of ACM Multimedia’95, 1995.

[2] 園田智也，後藤真孝，村岡洋一，“ＷＷＷ上での歌声による曲検索システム”，電子情報通信学会論文誌 Vol. J82-D2, No.4, pp. 721-731, 1999.

[3] 中島安貴彦,"MIDI バイブル Ｉ ―MIDI1.0 規格 基礎編", リットーミュージック，1997.

[4] Hevner, K., "Expression in music: A discussion of experimental studies and theories", Psychological Review, Vol. 42, pp. 186-204, 1935.

[5] 新版みんなで歌おう，株式会社日本標準，1990.

[6] 簡易ピアノ伴奏によるこどもの歌名曲アルバム，ドレミ学報楽譜出版社，1989.

[7] TV こどものうた，ドレミ学報楽譜出版社，1989

[8] [Online],http://www.kawai.co.jp/cmusic/products/scomwin.htm，スコアメーカー，（株）河合楽器製作所.

[9] [Online],mml2midiCompiler,http://www.platz.jp/~mml2mid/

[10] 橋口　博樹, 西村　拓一, 張　建新, 滝田　順子, 岡　隆一, モデル依存傾斜制限型の連続 DP を用いた鼻歌入力による楽曲信号のスポッティング検索, 電子情報通信学会論文誌 Vol.J84-D2 No.12 pp.2479-2488, 2001.

[11] 城風敏彦, 牧野正三, 城戸健一,”発話全体の連続性を考慮した基本周波数の検出”, 電子情報通信学会論文誌,Vol.J79-D2 No.9 pp.1537-1539, 1990.