音楽情報科学 24－4
(1998. 2. 19)

WWW 上での歌声による曲検索システム

園田 智也　　後藤 真孝　　村岡 洋一

早稲田大学 理工学部

〒169-8555 東京都新宿区大久保 3-4-1

TEL:03-3209-5198

E-mail:{sonoda, goto, muraoka}@muraoka.info.waseda.ac.jp

あらまし　本稿では、WWW上で動作する、歌声の旋律からその曲のタイトルを検索するシステムについて述べる。歌声による検索では、入力の旋律情報（音高・音長）が正確とは限らないため、閾値によってそれらを粗い旋律情報に変換したものを検索キーとし、データベースの曲とのマッチングを行なう。しかし、このための適切な閾値の設定は難しく、特に音長情報においては、有効な検索キーを得ることが困難であった。また、粗い旋律情報では正答の絞り込みも難しい。そこで、本研究では（1）有効な検索キーを得るための最適な閾値を設定する手法、（2）データベースの曲から正答の曲の候補を精度良く絞り込むためのマッチング手法の2つを提案することで、従来手法よりも正答率の高い検索を実現し、WWW上で複数の利用者が活用できるシステムを構築できた。

キーワード　音楽データベース、曲検索、メロディ検索、WWW アプリケーション、Java アプリケーション

A WWW-based Melody Retrieval System

Tomonari Sonoda　　Masataka Goto　　Yoichi Muraoka

School of Science and Engineering, Waseda University
3-4-1 Ohkubo Shinjuku-ku, Tokyo 169-8555, JAPAN.

Tel:03-3209-5198

E-mail:{sonoda, goto, muraoka}@muraoka.info.waseda.ac.jp

Abstract　This paper describes a WWW-based melody retrieval system which takes a sung melody as a query and retrieves the song name from a music database. In previous work, pitch information was mainly used as a clue while span information was not used effectively, and it was difficult to improve the matching accuracy by using only pitch information. We therefore propose the following two methods for effective matching. The first method can obtain the maximum quantity of information from pitch and span, and the second one reduces the answer candidates effectively. With these methods, we achieved higher matching accuracy than previous methods.

key words　music database, song retrieval, melody retrieval, WWW application, Java application

1 はじめに

本稿ではWWWブラウザ上で歌声を入力し、ネットワーク経由で曲データベース(**DB**)サーバからその曲のタイトルを検索するシステムについて述べる。

WWW上には多くの曲DBが公開されているが、これまで、それらに対する主な検索手法は歌詞などを文字で入力して行なうものであった。しかし、本来は耳で聞いて記憶している旋律を検索に用いる方がより直感的である。そこで、本研究では歌声の**旋律情報**(音高と音長の2つの属性値を持つ音符の系列)を用いた検索をWWW上で実現することを目的とする。

WWW上で歌声による曲検索システムを構築するには、様々な利用者に対して、精度良い検索を行なう必要がある。また、ネットワークの負荷を軽減するため、転送するデータサイズは小さい方が望ましい。

従来の曲検索システム[1]~[6]はいずれも1台の計算機上で実装され、ネットワーク上での利用は考慮されていなかった。歌声の旋律情報は、DB中の曲のものとは正確に調・テンポが一致するとは限らないため、[2]~[6]では、旋律の音高や音長系列を音符間の**相対音高差・相対音長比系列**に変換してから、検索に用いていた。さらに、利用者の記憶違いや歌唱能力による誤差を許容するため、相対音高差に対しては、前音から「上がった、同じ、下がった」、相対音長比に対しては、「長くなった、同じ、短くなった」などのように、**粗い精度の相対値**に変換したものを検索キーとしていた。ここで、検索キーを生成する際に、ある範囲の値を同一の値とみなすために、適当な閾値を利用していたが、以下の問題があった。

1. 有効な検索キー生成のための適切な閾値の設定が難しかった。特に音長に対しては設定が困難であった。

2. また、粗い相対値を検索キーとした検索では、DB中の曲も同様に粗い相対値に変換したため、複数の曲で同様の旋律パターンを持つことがあり、正答の曲が精度良く絞り込めないことがあった。

以上の問題のため、実際には音長が有効に用いられていなかった。さらに、粗い音高情報のみでは正答の絞り込みが難しく、WWW上で利用できるほど精度の良い検索を行なうことが困難であった。

そこで、本研究では以下の2つの手法を提案することで、これらの問題を解決する。

1. 曲DB中のすべての曲中に出現する音高・音長の分布を利用して閾値を決定することで、最も効率的に正答を絞り込む検索キーを生成する手法。

2. マッチングに用いる情報の精度を徐々に向上させながら検索を行なうことで、正答の曲を絞り込むマッチング手法。

以上の手法により、音高・音長の両者を有効に用いた精度の良い検索を実現することができた。また、歌声の膨大な音響信号を少量の旋律情報に変換するために、WWWブラウザのプラグインを利用することで、ネットワークにかかる負荷の少ない検索システムを構築することができた。

2 WWW上での曲検索システム

本システムの概要を図1に示す。本システムはサーバ・クライアント型のシステムであり、利用者はWWWブラウザのクライアント側で曲のメロディ(旋律)を歌うことで、サーバの保有するDBの曲の中から、最も似ている旋律を持つと解釈された曲のタイトルを得ることができる。

以下ではサーバ・クライアントそれぞれの処理の概要について述べる。

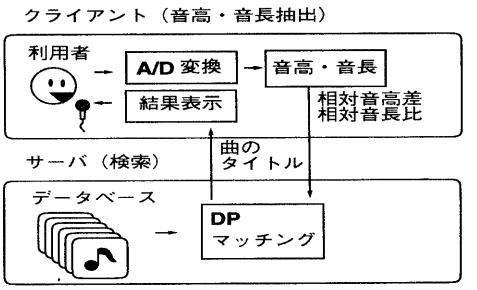

図1: システムの概要

2.1 クライアント（音高・音長抽出）

クライアントにおける処理の中心となる「音高・音長の抽出」について述べる。クライアントは、歌声の音響信号のA/D変換後、まず、処理の単位時間(フレーム、8msec)ごとに有声音の判定を行なう。次に、有声音のフレームを利用し、各音符の音高・音長の同定を行なう。最後に、音高・音長の系列を相対音高差・相対音長比に変換してサーバに転送する。このように、膨大な音響信号データを少量の旋律情報に変換することでサーバ・クライアント間のネットワークの負荷が小さいデータ転送を実現できる(この実装にはWWWブラウザのプラグインを利用する)。最後にサーバから検索結果であるタイトルを受け取るとWWWブラウザ上に表示を行なう。

クライアントの各処理を以下で示す。

- **歌声入力**
  歌声の入力では、利用者はマイクを使用し、自由な音高、自由なテンポ、自由な歌い出しで入力することが可能である。歌い方は、各音符が、無声音かつ破裂音で始まり 'a' の有声音で終るものに限定する（例：ta,cha,pa)。この形式により、利用者に大きな制約をかけずに、システムでは安定して旋律情報を抽出できた。

- **有声音の判定**
  歌声の音響信号の振幅が、一定値以上とき、有声音と判断する。

- **音高・音長の同定**
  連続する有声音のフレームのうちの一番最初を音符の発音時刻とし、各音符を発音時刻によって区切る。音符の発音時刻と次の音符の発音時刻との時間差（フレーム数）を音長とする。音長として定められた区間における各フレームごとのピッチを FFT (サンプリング周波数 8 kHz, 512 点) により求め、その最大値を音高とする（予備実験から最大値の方が平均値よりも精度良くピッチを同定できた)。

2.2 サーバ（検索）

サーバは曲 DB を保有しており、曲の「検索」処理を行なう。検索には DP マッチングを用い、クライアントからの入力の系列と DB 中の各曲中の系列との距離を求める。マッチングの結果、入力の系列と最も距離が近い系列をもつ曲から順に正答の候補とし、クライアントにそのタイトルのリスト（曲数は任意に設定できる）を送信する。

サーバの各処理を以下に示す。

- **曲 DB**
  DB は歌声によって作成できるのが望ましい。歌声で DB を作成する利点として、楽譜のない曲でも容易に DB に追加可能であることが挙げられる。そこで、本システムでは DB を歌声で作成し、歌声の入力と同様の方法で、各音符を相対音高差・相対音長比の系列として保存している。

- **DP マッチング**
  DP マッチングでは、入力と DB 中の曲との間で、系列間の距離を求める。マッチングには、音高・音長に対しての粗い精度の相対値を用い、比較する相対値の差を DP マッチングにおけるペナルティとし、その合計を系列間の距離とする。

3 実現上の課題と解決法

以上のシステム全体の検索精度に大きく影響を与えるのがサーバの DP マッチングに利用する情報の精度である。従来研究 [2]～[6] では、いずれも適当な閾値を利用して、マッチングに用いる相対音高差・相対音長比を粗い精度の相対値に変換して検索キーとしていた。例えば、[6] では、音高において、前音より「上がった、同じ、下がった」ということを表す値を U,E,D のように表し、「ドレミミレド」を「XUUEDD」などと変換したものを用いた（X は最初の音なので相対値がないことを示す)。しかし、適切な閾値を設定することは難しく、特に音長に対しては有効な検索キーを生成することができなかった。また、粗い精度の相対値では正答を絞り込むことが困難であった。

以下、**3.1, 3.2** では閾値の決定問題とその解決法を述べる。また、**3.3, 3.4** では正答の絞り込み問題とその解決法を述べる。

3.1 検索キー生成のための閾値決定問題

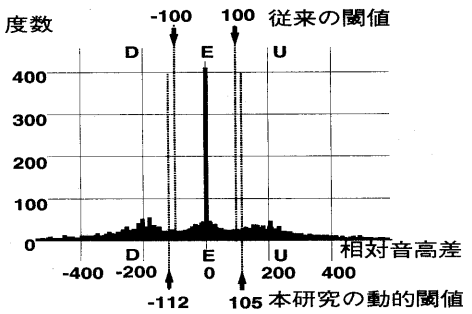

図 2: 相対音高差の分布 (200 曲) と閾値

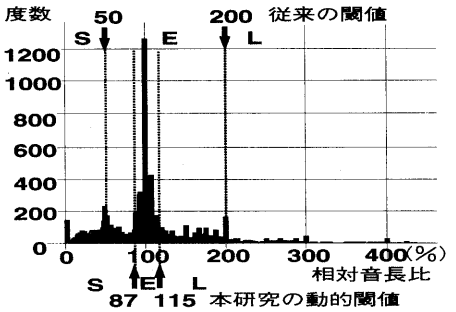

図 3: 相対音長比の分布 (200 曲) と閾値

図 2、3 はそれぞれ、歌声によって作成した DB の曲（200 曲）中に出現するすべての相対音高差・相対音長比のヒストグラムである。相対音高は半音の差

が 100 となるように正規化しており、相対音長は比をパーセンテージで表現している。ここで、図中に従来研究 [2] で用いられた検索キー生成のための閾値を示す（図中の動的閾値については **3.2** で述べる）。[2] では、相対音高差に関しては、前の音符から半音の幅の差を閾値としていた。また、相対音長比に関しては、1/2 倍 (50%) と 2 倍 (200%) を閾値としていた。ここで、閾値によって変換される粗い精度の相対値を表す記号として、音高には「上がった、同じ、下がった」を表す U,E,D を用いる。音長には「長くなった、同じ、短くなった」を表す L,E,S を用いる。これらの図を見ると、音高に関しては U,E,D それぞれにほぼ均等に変換されていた。このため、DB 中の曲の系列に出現する U,E,D がほぼ等しい確率で出現し、検索キーの 1 つの音符ごとに絞られる曲の数は 1/3 ずつになっていくことが期待される。一方、音長に関しては、多くのものが E に分類され、L,S に分類されるものは比較的少なかった。このため、DB 中の多くの曲の音長の系列は、E が L,S に比べ多く出現する系列に変換されていた。この結果、複数の曲で同様のパターンが出現する確率が高くなり、正答の曲を絞り込むことが難しかった。

以上のように、DB 中の曲の系列パターンを考慮に入れた適切な閾値の設定は難しく、特に音高と比較して、音長は精度の良い検索を行なうための閾値の設定が困難であった。また、曲 DB によっては、各曲の相対音高差・相対音長比の分布に偏りのある場合が考えられるため、曲 DB の性質に応じた閾値の決定法が必要であった。

3.2 閾値決定問題に対する解決法

本研究では DB 中の曲のすべての相対音高差・相対音長比に対して、変換される値のカテゴリに含まれる相対値の合計度数が、カテゴリ間でなるべく均等となるように、閾値を設定する手法を提案する。

図 2、3 に本手法の閾値を示す。DB 中の曲の相対音高差・相対音長比は相対値を本手法の閾値によって変換すると、DB 中の変換された系列に出現する相対値（U,E,D/L,E,S）は、出現確率は等しくなる。検索時に、入力の相対音高差・相対音長比の系列もこの閾値により変換することで、入力 1 音符ごとに、正答の曲が均等に絞られていく。

また、この閾値決定法の応用として、従来 3 段階にしか分類されなかった情報の粗さの精度が、自由に設定できることが可能となる。例えば 5 段階に情報の粗さを設定する場合には 5 つのカテゴリに分類される相対値の度数が均等に分類されるように閾値を決定すれば良い。これにより、3 段階の粗さの精度では絞り込めなかったような曲 DB に対しても、より細かい粗さの精度で閾値を設定することで、正答の曲をより精度良く絞り込むことが可能となる。

この閾値の決定手法は DB 中の曲すべてに出現する音高・音長の相対値の分布から動的に定めているので、従来の静的な決定法に対して、**動的閾値決定法**と呼ぶことにする。また、本稿では、従来の閾値を**静的閾値** と呼ぶことにする。具体的な動的閾値の決定法については **4** で述べる。

3.3 正答の絞り込み問題

粗い相対値を検索時のマッチングに用いると、曲 DB 中の多くの曲で同様の系列パターンが出現する確率が高くなり、正答の絞り込みが難しくなる（図 4）。

この問題を解決するために、より精密な精度での音高・音長の相対値を用いる必要がある。本手法では動的閾値決定法を提案したため、従来手法とは異なり、より精度の良い検索が行なうことができるようになった。しかし、粗さの精度がより細かくなると、利用者の歌声の誤差に影響を受けるようになる。そこで、入力の誤差を考慮しながら、精度の高い検索を行なう手法が必要となる。

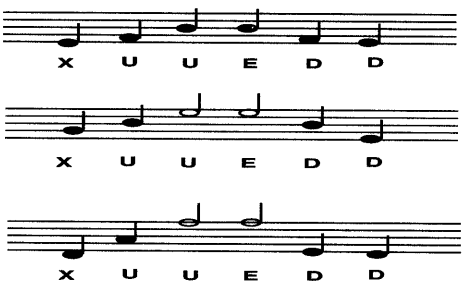

図 4: 粗いマッチングに見られる現象

3.4 正答の絞り込み問題に対する解決法

本研究では、検索に用いる情報の粗さの精度を徐々に向上させながらマッチングを行なう手法を提案する。本手法では、まず、大まかな精度（例えば 3 段階）の相対音高差・相対音長比を利用し、 DP マッチングを行なう。その結果、DB 中の曲で入力との距離が近いと解釈された曲の上位に着目する。ここで、その精度で正答を絞り込めていたかどうかの判定をし、絞り込めていなかった場合にのみ、より細かい精度（例えば 9 段階）の相対値を利用したマッチングを、上位の曲に対して繰り返す（図 5）。

これは、粗い精度のマッチングでは、正答は絞り込むことが難しいが、正答の曲は上位に含まれるという性質を利用した手法である。上位に含まれた曲を対象とした、より精密なマッチングを用いることで正答の曲が精度良く求められる。

実際には、十分な分割数の閾値を用意することで、複数の閾値を持つことを回避できる。例えば、$3^3 = 27$ 段階の閾値を予め用意することで、3 段階、9 段階、27 段階の閾値を別々に用意する必要がない。

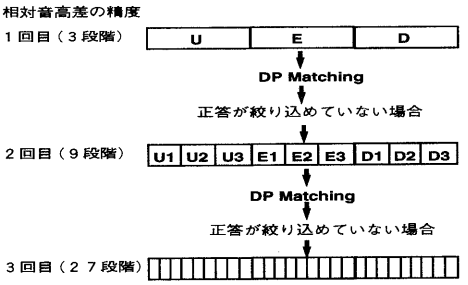

図 5: Coarse-to-Fine マッチング法の例

このように精度を徐々に高めながらマッチングを行なう手法は、画像のマッチング処理などで用いられており、本研究ではその概念を曲検索システムに応用することで精度良い検索を実現する。そこで、この手法を曲検索における **Coarse-to-Fine マッチング法**と呼ぶことにする。具体的な Coarse-to-Fine マッチング法の手順については、**5** で述べる。

4 動的閾値決定法

動的閾値決定法の手順と、使用法について述べる。

4.1 動的閾値決定法の手順

動的閾値 DT の決定法は以下のようになる。

1. DB 内の全曲中の音高、音長の相対値で図 2、図 3のようなヒストグラムを作成する。
2. ヒストグラムの総度数を Sum とし、カテゴリ数を $CategoryNum$ としたとき、1 つのカテゴリ内の合計度数の期待値を $M = Sum/CategoryNum$ とする。
3. 各カテゴリの合計度数が均等に M に近くなるようにヒストグラムを分割し、その各境界線を閾値 DT_j $(0 \leq j < CategoryNum - 1)$ とする。

このようにして決定した動的閾値を利用することで、歌声の入力と DB 中の曲すべてに出現する音高・音長の相対値の粗い相対値が得られる。以下では、動的閾値の利用について具体的に解説を行なう。

4.2 動的閾値を用いた検索キーの生成

相対音高差の系列 Q_p、相対音長比の系列 Q_s に対する動的閾値 DT_p, DT_s を利用し、それぞれの粗い相対値情報の系列 R_p, R_s を求める。

$DT_n(j)$ $(0 \leq j < CategoryNum - 1)$ $(n = p$ または $n = s)$ によって変換される $CategoryNum$ 段階の粗い相対値 c_k $(0 \leq k < CategoryNum,\ c_k < c_{k+1})$ の集合を C とする。長さ m の入力系列 Q_n を

$$Q_n = q_0\ q_1\ q_2\ \cdots\ q_{m-1}$$

で定義するとき、Q_n に出現する各音符の相対値 $q_i (0 \leq i < m)$ を $CategoryNum$ 段階の粗い相対値情報 r_i $(\in C)$ に変換する際の条件は、以下のようになる。

$$r_i = \begin{cases} c_0 & (q_i < DT_n(0)) \\ c_k & (DT_n(k-1) \leq q_i < DT_n(k)), \\ & 1 \leq k < CategoryNum - 1 \\ c_{n-1} & (DT_n(CategoryNum - 1) \leq q_i) \end{cases}$$

この変換により、Q_n は粗い相対値情報の系列

$$R_n = r_0\ r_1\ r_2\ \cdots\ r_{m-1} \quad (r_i \in C)$$

に変換される。マッチング時には、DB 中に出現する音高・音長に対しても同様の処理を行なえば良い。

5 Coarse-to-Fine マッチング法

Coarse-to-Fine マッチング法の処理手順を述べる。

歌声入力の相対音高差・相対音長比の系列 K_p, K_s と DB 中の曲の相対音高差・相対音長比の系列 S_p, S_s との間で、DP マッチングを行なう際、両者のそれぞれの系列を G_i 段階の粗い精度の相対値 $KR_p(i), KR_s(i)$ と $SR_p(i), SR_s(i)$ に変換することを考える（ただし、$i = 0, 1, \ldots, m,\ G_i < G_{i+1}$とする）。この変換には、動的閾値 $DT_p(i), DT_s(i)$ を用いることにする。このとき、曲 DB から、歌声の入力の旋律に近い旋律を持つ曲を順に N 番目まで絞り込む検索を Coarse-to-Fine マッチング法によって行なう。ここで、用いる相対値の粗さの度合は m 段階に向上させるとする。以下にその処理手順を示す。

1. $i = 0$ とする。
2. 動的閾値 $DT_n(i)(n = p$ または $n = s)$ によって K_n と S_n を、それぞれ G_i 段階の粗い相対値の系列 $KR_n(i)$ 、$SR_n(i)$ に変換する。

3. $KR_n(i)$ と $SR_n(i)$ の間で DP マッチングを行ない、両者の距離 D を求める。

4. マッチングの結果、得られる各曲の D の小さいものから順に正答の候補の上位とし、上位から N 番目までの曲に着目する。それらの D がすべて異なっている場合は絞り込みが完了しているとし、処理を終了する。複数の曲の D が同じ値である場合は、正答を絞り込めていない状態とみなす。そこで、N 番目までの曲の系列 $SR_n(i)$ と入力の $KR_n(i)$ に対して、より精度の良い閾値 $DT_n(i+1)$ を用い、G_{i+1} 段階の情報の系列に変換する（$G_i < G_{i+1}$）。

5. 以下、$i+1$ を新たな i とみなし、絞り込みが完了するか、または、$i = m-1$ となるまで 3. 4. の処理を繰り返す。

6 実験と考察

本稿で提案した動的閾値決定法の評価、Coarse-to-Fine マッチング法の評価、検索の正答率の評価、処理時間の評価に関する実験を行なった。以下に実装環境と各実験、考察を述べる。

6.1 実験環境

サーバを Sun Ultra Enterprise 3000 (UltraSPARC,167MHz) 上に実装した。クライアント は WWW ブラウザとして Netscape Navigator を利用し、JAVA のアプレットとプラグインで実装したものを SGI indigo2 Impact(R4400,250MHz) 上で実行した。このとき、8 kHz でサンプリングされた音声を 1 フレーム 64 点 (8msec) でシフトしながら、512 点での FFT を行なうことで歌声の音響信号を処理した。

6.2 閾値による情報量の比較

動的閾値が静的閾値よりも情報量の大きい検索キーを生成することを確認する評価実験を行なった。

6.2.1 実験

動的閾値、静的閾値のそれぞれで音高・音長情報を粗い情報（3 段階）に変換する。曲 DB 中のすべての音符に対する各カテゴリ内の音符の出現確率 p_i から情報エントロピーを計算する。情報エントロピーを求める式は以下のものを用いた。

$$-\Sigma p_i \log_2 p_i$$

結果を表 1 に示す。なお、表の一番右の欄の「最大値」とは、3 つのカテゴリに分類される情報が持つ最大の情報エントロピーの値を表す（計算式: $3 * (1/3) log_2 3 = 1.5850$）。

	音高	音長	最大値
動的閾値	1.5850(bit)	1.5846(bit)	1.5850(bit)
静的閾値	1.5836(bit)	1.2025(bit)	1.5850(bit)

表 1: 閾値による情報エントロピーの比較

6.2.2 考察

静的閾値による分類では音長情報は明らかに情報量が少ないが、動的閾値による分類では音高情報も音長情報もほとんど最大値に近付いている。この結果より、動的閾値が有効な検索キーを生成するのに有効な手段であることが確かめられた。

6.3 動的閾値決定法の性能

実際の検索での、動的閾値決定法による正答率の向上を確認する評価実験を行なった。

6.3.1 実験

静的・動的閾値による検索精度の比較を行なった。様々なジャンル（日本・西洋のポップス、童謡、民謡、演歌など）からなる 200 曲の DB （総音符数 16718 個）に対し、12 人（男 8:女 4）の被験者の音声によって、音高のみ、音長のみ、および音高・音長組み合わせによる検索をそれぞれ 112 回行なった。閾値の条件以外は同条件で行ない、粗い旋律情報のカテゴリ数は音高、音長ともに 3 段階とし、Coarse-to-Fine マッチング法は用いていない。静的閾値は、音高に関しては半音以内の音高差を「同じ」とみなし、音長に関しては 1/2 より大きく 2 倍未満の音長比を同音長とした [2]。入力を 8 秒間以内で限定すると、入力平均音符数は 18.5 で あった。正答率を厳密に 1 位に絞られて正答した場合、3 位以内に入っていた場合で評価した結果を表 2、3 に示す。

6.3.2 考察

音高での検索は、それほど違いが生じなかったが、音長は明らかに正答率が向上したことが確認できた。音高・音長を組み合わせた検索も向上していることから、本手法が有効であることが確かめられる。

	音高検索	音長検索	組合せ検索
正答率	47.3%	13.4%	90.2%
3 位以内	65.2%	25.0%	95.5%

200 曲に対して 112 回検索

表 2: 静的閾値を利用した従来の検索

	音高検索	音長検索	組合せ検索
正答率	49.1%	35.1%	97.3%
3 位以内	67.0%	50.1%	99.1%

200 曲に対して 112 回検索

表 3: 動的閾値を利用した検索

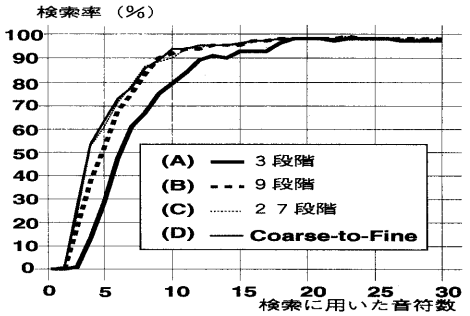

図 6: Coarse-to-Fine と各段階の粗さでのマッチング

6.4 Coarse-to-Fine マッチング法の性能

Coarse-to-Fine マッチング法で 3 → 9 → 27 段階に音高・音長の精度を変化させた検索行ない、その検索の正答率を、各段階で用いた精度でのマッチングの結果 (Coarse-to-Fine マッチング法を用いないもの) と比較し、Coarse-to-Fine マッチング法の性能について考察する。

6.4.1 実験

次の 4 つの条件 **A** ～ **D** において検索に用いる音符数を 1 ～30 に変化させ、正答率を比較した。**A,B,C** については Coarse-to-Fine マッチング法を用いず、**D** のみに用いている。Coarse-to-Fine マッチング法においては、上位 1 曲のみに着目し、2 位の曲との差が生じなかったときに精度をあげた処理をした。6 に示す。ここで、正答率は厳密に正答が 1 曲に絞り込まれているもので評価した。

- **条件 A**
  動的閾値によって 3 段階の粗い相対値情報に変換した音高・音長を用いた検索。
- **条件 B**
  動的閾値によって 9 段階の粗い相対値情報に変換した音高・音長を用いた検索。
- **検索 C**
  動的閾値によって 27 段階の粗い相対値情報に変換した音高・音長を用いた検索。
- **検索 D**
  Coarse-to-Fine を用い、音高・音長を動的閾値によって 3 → 9 → 27 段階に変化させながら検索した結果。

6.4.2 考察

Coarse-to-Fine マッチング法での検索は、27 段階の粗い旋律情報を利用した検索と正答率がほとんど一致した。これは、Coarse-to-Fine マッチング法の精度が最終的に 27 段階のものを用いたためだと考えられる。しかも、音符数が 10 をこえる検索では、8 割以上の曲で 3 段階の精度でマッチングが終了していた。このため、多くの曲で、精度向上の繰り返し処理をせずに、検索を終了できたことが確認できた。この実験では DB サイズが小さかったため、9,27 段階での検索と Coarse-to-Fine を用いた検索での正答率の評価は難しい。人工的にエラーを投入した正答率を計測してもほとんど差が生じなかった。

6.5 本システムの検索精度

本研究で提案した動的閾値決定法と Coarse-to-Fine マッチング法を組み合わせ、正答率の評価を行なう。

6.5.1 実験

歌声の入力データ、曲 DB 、入力時間は **6.3** と同様の条件で、3 → 9 → 27 段階 の Coarse-to-Fine マッチングを行なった (各段階の音高・音長の精度で上位 3 位以内が 1 つに絞られていない場合にのみ次の段階の精度でマッチングを行なうようにした)。なお、動的閾値は 27 段階のものを作成した。以上の条件で、音高のみ、音長のみ、音高・音長を組合せた検索を行なった。結果を表 4 に示す。

6.5.2 考察

音高・音長ともに、従来 3 段階のみで検索していた表 2 の正答率より 30 ～40 % の向上が見られた。

	音高検索	音長検索	組合せ検索
正答率	84.8%	74.1%	98.2%
3 位以内	89.3%	86.6%	99.1%

200 曲に対して 112 回検索

表 4: 動的閾値と Coarse-to-Fine を用いた検索

6.6 処理時間・データサイズ

各処理に要する時間を計測した。また、ネットワークにかかる負担を測定するために、クライアントからサーバに転送されるデータサイズについて計測した。

6.6.1 実験

システムにおける各処理時間を以下に示す。

- **動的閾値の生成時間 (6.41 sec)**
  サーバの実装環境で 200 曲 (総音符数 16718 個) の DB に対する動的閾値の生成を 100 回行なった平均値。動的閾値は 27 段階の粗い相対値を生成するものを利用している。

- **検索処理時間 (16.38 sec)**
  サーバの実装環境で 200 曲 (総音符数 16718 個) の DB に対する 112 回 (総音符数 3708 個) の検索を行なった平均値。動的閾値と Coarse-to-Fine (3,9,27 段階) マッチングを用いた検索を行なった結果である。

また、クライアントからサーバに転送されるデータのサイズを計測したところ、8 秒間の入力における最大値は 420 bytes、平均値は 274.5 bytes となった。

6.6.2 考察

動的閾値の生成と検索には、DB 中のすべての曲の音符数の合計に比例する処理時間が必要である。動的閾値の生成にかかる時間はそれほど大きく、将来的に、1 万曲規模の DB に対しても、実験に用いた DB と 1 曲あたりの音符数 (本 DB では 83.6 個) が変わらないならば、約 50 倍の時間 (5〜6 分) で終了する。

検索時間に関しては、現在の実装では 200 曲に対して十分な時間で行なえた。今後は 1 万曲規模の DB に適用できるように、マッチングアルゴリズムの改良が必要となる。

クライアントからサーバに対して転送されるデータは、非常に小さく、入力時間を、仮に数倍長くしてもネットワークにかかる負荷や転送時間における問題は発生しないことが確認できた。

7 まとめと今後の課題

本稿では WWW 上で複数の利用者が活用できる歌声による曲検索システムを提案した。また、システムのマッチング精度を向上させるために、動的閾値決定法を提案し、検索キー生成のための最適な閾値を得ることができた。さらに、Coarse-to-Fine マッチング法を導入することで、正答の曲の候補を効果的に絞り込むシステムを実現した。今後の課題としては、以下のことが考えられる。

1. DP マッチングは DB 中の音符の総数に比例した計算時間を伴う。探索する曲数を減少させるためにアルゴリズムを改良し、大規模な曲 DB に対しても高速に精度良く検索が可能となるシステムの開発を目指したい。

2. 現在の実装では、予め用意された曲 DB に対する検索のみ可能である。しかし、WWW 上には MIDI などの様々な曲データが公開されている。それらのデータを収集する音楽版 WWW Robot を開発し、世界中の WWW 上の 曲 DB に対して検索を行うシステムの構築を目指したい。

参考文献

[1] 貝塚 智憲, 後藤 真孝, 村岡 洋一: 歌声の旋律情報と歌詞情報をキーとした曲検索システム, 54 回情報学会全国大会, 1997.

[2] 蔭山哲也 , 高島洋典 : ハミング歌唱を手掛かりとするメロディ検索, 電子情報通信学会論文誌 D-II Vol.J77-D-II No.8, pp.1543-1551, 1994.

[3] T.Kageyama, K.Mochizuki, Y.Takashima : *Melody Retrieval with Humming*, ICMC Proc., pp.349-351, 1993.

[4] 蔭山哲也 : 音高・音長情報を利用したメロディ検索, 45 回情報処理学会全国大会, 1-357, 1992.

[5] 蔭山哲也 , 島津秀雄 , 高島洋典 : メロディ検索 - ハミングで音楽 *DB* を検索する , 43 回情報処理学会全国大会, 4-149, 1991.

[6] Asif Ghias, Jonathan Logan : *Query By Humming – Musical Information Retrieval in an Audio Database*, ACM Multimedia 95, Electronic Proc., 1995.