DP マッチングを用いたオーディオ・パターンマッチングの特性改善

2Q－5

松井　唯史　　甲藤　二郎

早稲田大学大学院理工学研究科

1. はじめに

インターネットや携帯電話を利用した音楽配信サービスの話題を耳にするようになって久しい。通常、MP3 に代表されるオーディオ符号化技術は、MDCT、ステレオ符号化などを用いることで実現されているが、現在用いられている符号化手法のみでは、音質を保つことが限界に達しつつある。しかし、既存のオーディオ符号化方式では、携帯電話などの低帯域なネットワーク環境ではダウンロードしながらリアルタイムに音楽を楽しむことは難しく、またフラッシュメモリのような少ない容量のメモリに多くの音楽を記録しておくことも困難なため、更なる冗長を削減することが求められている。

MPEG ビデオの符号化方式と比較して、主に線形予測を用いている MPEG オーディオでは、時間軸方向の相関を利用した冗長削減の割合が非常に小さいと考えられる。しかし、特に DTM によって製作された音楽に関しては、同じパターンが何度も繰り返される可能性が高い。本稿では、そのような同一パターンを音楽ソースの中から探索し、重複するパターンの情報量を削減するアルゴリズムについて提案する。

2. オーディオ・パターンマッチング [1]

2.1 オーディオ・パターンマッチングの概要

オーディオ・パターンマッチングの概念図を図 1 に示す。その特長は、音楽信号の中から類似パターンを探し出し、繰り返されるパターンは一度しか記録しないようにすることにある。各パターンは、必要に応じて更に MP3 などの既存の符号化アルゴリズムで圧縮することができるようになっており、既存の符号化技術と共存することが可能である。ファイルヘッダには再生順序情報があり、再生アプリケーションでは、その情報をもとに各パターンを途切れ目のないよう滑らかに再生するため、原信号との聴感上の差は感じられないようになっている。

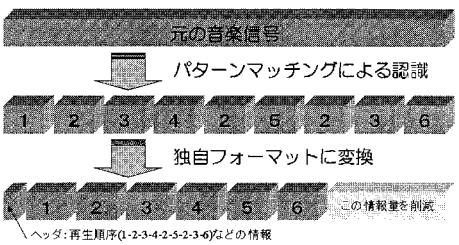

図1 オーディオ・パターンマッチングの概念図

"Performance Improvement of Audio Pattern Matching Using DP Matching"

Tadashi Matsui, Jiro Katto

Graduate School of Science and Engineering, Waseda University

2.2 予備実験

オーディオ・パターンマッチングは、ファイルフォーマットさえ決めることができれば、波形編集ソフトなどを用いて手動で分割を行い、異なるパターンのみを抜き出してくることによってファイルを作成することも可能である。そのような手法を用い、市販されている音楽がオーディオ・パターンマッチングによってどの程度情報を削減できるかの予備実験を行ってみた。その結果を表 1 に示す。

表1 予備実験の結果

	音楽 A	音楽 B	音楽 C	音楽 D	音楽 E	音楽 F
元の PCM ファイル	55,364,632	47,859,428	41,190,620	53,285,748	58,455,252	65,973,644
パターンマッチング後	52,219,489	42,618,129	32,788,645	40,900,815	42,953,548	38,738,426
情報削減率(%)	5.68	11.0	20.4	23.2	26.5	41.3
パターンマッチング+MP3	4,736,413	3,865,601	2,936,552	3,710,154	3,896,196	3,513,736
平均ビットレート(kbps)	121	114	102	98.3	94.0	75.2

情報削減率,平均ビットレート以外の単位は bytes

音楽 A～音楽 C は一般的な日本のポップス、音楽 D～音楽 F は同じパターンが繰り返される確率がより高いと考えられるダンス系の音楽である。この予備実験により、A～C では 5%から 20%、D～F では 23%から 41%の情報を削減できることが確かめられた。 また、主観評価ではあるが、この予備実験による音質劣化は認められなかった。

この予備実験の結果に加え、更に MP3 を用いて圧縮した結果も併せて示している。MP3 はビットレート 128kbps でエンコードを行ったが、その平均ビットレートはいずれも 128kbps を下回っており、パターンマッチングの効果が生かされていることが確認された。

2.3 従来方式

では、2.2 のシミュレーションと同様のことを信号処理で実現するためには、どのようにすればよいか。本研究室では、これまで次のようなアルゴリズムを提案してきた。図 2 に、そのフローチャートを示す。

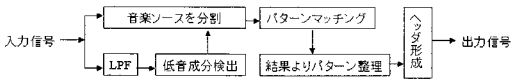

図2 全体のフローチャート

先ず、音楽信号を適当な長さに分割する必要がある。ここで、MP3 のようにある一定のサンプル数で音楽を分割してしまうと、分割されたブロックは音楽として意味のない長さになり、同じパターンを抽出することが難しくなる上、再生時にはパターンの途切れ目が不自然な音楽になってしまう。

これを回避するために、音楽信号にローパスフィルタを適用し、バスドラムなどのリズムを刻んでいる低音成分を抽出する。[2] 次に、抽出された成分のピークを探索し、ピーク間のサンプル数に忠実に音楽ソースを分割する。 このようにして

分割されたブロックは、一小節などといった音楽として意味のある長さになり、パターン抽出の難しさや再生時の不自然さは解消される。

次に、得られた各ブロックの中から、比較的サンプル数の近いブロック同士に対してのみマッチング処理を行う。マッチング処理が終了したら、類似ブロックが連続している箇所同士を同一パターンと決定し、整理されたパターンと演奏順序情報をファイルとして出力する。

従来方式では、パターンマッチング処理として、各ブロック同士を 1 サンプルずつ移動させながら差分をとるだけの移動マッチング法を用いてパターンマッチングを行っていた。しかし、特にアナログ的に編集して作られた音楽については、信号に時間的な非線形伸縮が生じていることが多く、聴感上は類似ブロックのように聴こえても、移動マッチングだけではそれを判断することは困難であった。

3 提案方式

3.1 DP マッチング

従来方式の問題点を解決するため、DP(Dynamic Programming) マッチング法を導入した。[3] これは、変形した二つの信号間のマッチングに特化したアルゴリズムで、音声認識や指紋照合などの分野で広く用いられている。この手法により、二つのオーディオ信号間に生じた時間的な非線形伸縮は吸収されると考えられる。

二つの信号間の距離を g(i,j)、局所距離を d(i,j)とするとき、DP マッチングを次の漸化式で与えた。置換誤りに相当する局所距離が 2 倍されているのは、どのような照合の仕方によっても局所距離の加算回数が一定となるようにするためである。

初期条件 : $g(0,0)=0 \quad g(i,0)=g(0,j)=\infty$ for i=1,2,・・・,I ,j=1,2,・・・,J

i=1,2,・・・,I, j=1,2,・・・,J

$$g(i,j)=\min \begin{cases} g(i-1,j) + d(i,j) \\ g(i-1,j-1) + 2d(i,j) \\ g(i,j-1) + d(i,j) \end{cases}$$

オーディオ信号の場合、d(i,j)は二つの信号のサンプル間の距離などとし、最後に計算される g(I,J)を局所距離の加算回数で正規化した値を求める。通常、その値が最小となる組み合わせをマッチング結果とする。

3.2 実験結果

2.3 の手法を用いて三種類の音楽から抽出された、それぞれ 20 個のブロックを用いて実験を行った。これらのブロックは、10 通りの類似した組み合わせを含んでいる。このとき、全てのブロックの組み合わせは ${}_{20}C_2=190$ 通りとなり、その全ての組み合わせについて DP マッチングを行う。ある一つのブロックに対して、もう一つのブロックとの距離が最小となる組み合わせを類似と判断し、それ以外の組み合わせは非類似と判断する。その結果が聴覚による判断と一致したものを正解とした。その正解率を表 2 に示す。

表2 実験結果

	移動マッチング	DP マッチング
音楽1	97.4%	99.5%
音楽2	95.3%	96.3%
音楽3	94.7%	100%

3.3 高速化手法

DP マッチングの導入により正解率は向上したが、パターンマッチングに必要な処理時間が大幅に増大したため、処理時間を短縮する手法についても検討する必要がある。ここでは、アルゴリズムの工夫やハードウェアの機能を利用した DP マッチングの高速化手法について述べる。

オーディオ信号のマッチングにおいては、極端な時間的伸縮が起きていることは通常あり得ないと考えられる。そこで、時間的伸縮の長さに制限を設けることで、g(i,j)の計算量を大幅に削減できる。更に計算量を削減するために、探索の途中で処理を打ち切ったり、あるいは階層的に探索することも、計算時間の観点からは有効である。もう一つの考え方としては、極力 DP マッチングを行わずに済むように、包絡線などから明らかに類似ではないブロック同士のマッチング処理を、前処理としてスキップする方法もある。但し、以上の処理によってマッチングの正解率が下がる恐れもあり、その評価を進めている。

一方、メモリの使い方にも工夫が必要である。一般的な音楽の場合、1 ブロックは 20000 サンプル以上にもなるため、計算した g(i,j)の値を全てメモリ上に置いておくと空きメモリを食い尽くしてしまい、大幅にパフォーマンスが低下する。不必要になった g(i,j)の領域は順次開放していく必要がある。

また、最近の CPU は MMX や SSE といったマルチメディア信号処理を意識した拡張命令を搭載していることが多い。必然的に提案方式は並列処理が容易に行えるものであり、これら CPU 固有の SIMD 命令を用いることによっても高速化が可能である。

4. おわりに

DP マッチングを用いることにより、オーディオ信号の時間的な非線形伸縮を吸収し、パターンマッチングの正解率を高めることに成功した。現在では、類似するパターンを一つだけ残して他は削除してしまう手法を用いているが、今後は類似するパターンの差分信号のみを記録することにより、ロスレス符号化への適用などの拡張も検討する予定である。

参考文献

[1] 松井唯史,甲藤二郎 : "オーディオ・パターンマッチングに関する一検討", 2001 年 電子情報通信学会総合大会 D-14-11

[2] 後藤真孝,村岡洋一 : "音楽音響信号を対象としたビートトラッキングシステム",音楽情報科学 2-18, pp.45-52

[3] 中川聖一 著 : " パターン情報処理",丸善,1999,pp153-163