適正な音楽流通のためのインターネットを介したオーディオ電子指紋の超高速検出

代表研究者　井口　寧　北陸先端科学技術大学院大学情報科学センター准教授
共同研究者　Vijay K. Jain　南フロリダ大学電気工学科特別教授

1 はじめに

近年，Winny や WinMX など，インターネットでデジタル著作物を交換するソフトウェア(ファイル交換ソフト)が広く用いられ，デジタル著作物の著作権侵害が大きな社会問題となっている．このデジタル著作物を保護するための一手法として，電子指紋技術が注目されている．電子指紋は，音楽や画像などの周波数帯域ごとのエネルギー遷移など，著作物ごとの特徴量を計量することによって著作物を特定する技術であり，著作物のコピーコントロールを成し得る重要な技術である．ところで電子指紋検出は従来ソフトウェアで行われており，検出速度が低速であるため，ネットワークを介して流通するデジタル著作物の電子指紋の実時間検出は困難という問題があった．

電子指紋の計算には幾つかの方法が提案されているが，代表的なアルゴリズムは周波数ごとのエネルギー遷移などに差分計算を施し，およそ 1KB ほどのコンパクトな表現を生成する．同一の楽曲であっても，MP3 や AAC，WMA など圧縮方法が異なれば，僅かだが指紋も異なってしまうが，同じソースからはほぼ同一の指紋となるアルゴリズムである必要がある．従来の指紋生成アルゴリズムは，音楽ファイルの非可逆圧縮などに対する耐改変性を高めることに主眼を置いたアルゴリズムが大半である．

ところで，近年チップ内の回路をユーザーが自由に書き換え可能な素子(FPGA; Field Programmable Gate Array) の集積度が飛躍的に向上し，これらの書き換え可能な素子を用いて内部回路を動的に再構成しながら処理を進めるリコンフィギャラブル・コンピューティング(RC; Reconfigurable Computing)が注目されている．電子指紋の計算アルゴリズムは，多くの部分が固定小数点の加減算で行われており，このような RC に対する適合性は潜在的に非常に高いと考えられる．しかしながら，従来の電子指紋アルゴリズムはハードウェア実装を考慮して設計されておらず，一部の浮動小数点演算が全体のパフォーマンスを大きく下げている結果となっている．

そこで本研究ではハードウェア実装に適した，新しいオーディオ電子指紋検出アルゴリズムを提案する．このアルゴリズムは，Haar 基底の Wavelet 変換を用いる手法で，殆どの処理を固定小数点の加減算のみで行うことができ，ハードウェア実装に対する適合性が非常に高い．本アルゴリズムをリコンフィギャラブル・システムに適用した時の検出精度，誤識別率，検出速度，回路消費量などについて評価した．

本論文の構成は次の通りである．第 2 章では，オーディオ電子指紋の概要について紹介し，第 3 章ではハードウェア実装向け電子指紋検出アルゴリズムを提案する．第 4 章では，提案した電子指紋アルゴリズムの評価を行い，最後に第５章でまとめと今後の課題について述べる．

2 オーディオ電子指紋について

2-1 電子指紋の概要

オーディオ向け電子指紋アルゴリズムは多くの方法が提案されている[1-11,16-17]．電子指紋のアイディアは，音楽の特徴を数キロ bit の比較的コンパクトな表現(電子指紋)として取り出し，これを比較することによって音楽のソースが同一かどうか判定しようというものである[3,4]．通常，音楽ファイルをインターネットを介して交換する場合，MP3 や AAC，WMA など非可逆な音楽向けの圧縮アルゴリズムで圧縮し，受信側で伸張する過程となる．この場合，圧縮-伸張後の音楽ファイルはオリジナルの信号と全く同一になることは期待できず，圧縮-伸張過程が異なれば，伸張後の音楽ファイルを入力とする電子指紋も異なる結果となる．しかしながら，いずれの圧縮方法も人間の耳に違和感が残らないような圧縮を行っている．この点に注目し，電子指紋の演算過程で音楽の特徴量を抽出し，人間の耳で同じ楽曲と判断される場合にはできるだけ小さい差異を与える電子指紋を生成できれば，電子指紋によって楽曲を識別できる．電子指紋はビットパターンで表されるので，二つの電子指紋のハミング距離を求めることによって，楽曲の差異を判定することができる．

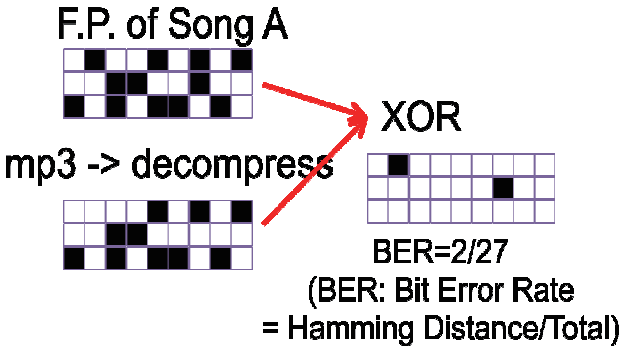

図 1 電子指紋の例（2 つの楽曲が同じ場合）

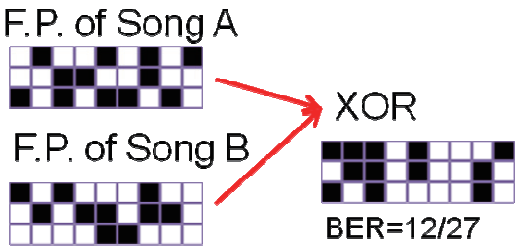

図 2 電子指紋の例（2 つの楽曲が異なる場合）

図 1 および図 2 に電子指紋の例を示す．同じ楽曲(Song A)をソースとする場合，ソースから直接生成された電子指紋は”F.P. of Song A”で示される．同時に Song A を非可逆圧縮-伸張した場合，通常伸張後の音楽ファイルは圧縮前の音楽ファイルと同一にはならない．このため，2 つの電子指紋は異なった出力となる．しかしながら，ソースが同じであるため，両者のビットパターンは類似しており，このハミング距離(XOR 演算後の”1”の数)は非常に少なくなる．本稿では，この差異を BER (Bit Error Rate) として表し，この場合，BER は 0 に近くなることが期待できる．

一方，図 2 では二つの違った楽曲から電子指紋を生成している．ソースが全く異なるので，生成される電子指紋も全くことなるビットパターンとなる．従って，両者の差異を表す BER は，全く相関が無く，およそ 0.5 になることが予想される．

2-2 先行研究

このような電子指紋を計算するためのアルゴリズムがこれまでに多く研究され，強度，利便性，耐雑音性，耐改変性などの観点から多くの方式が提案されてきた．Cano らはプロトタイプとなるオーディオ電子指紋の概要を提案した [3]．これから派生した幾つかのアルゴリズムが提案され，スペクトル扁平の特徴を利用するもの[4]，スペクトルのピークの特徴を利用するもの[5]，フーリエ係数の特徴を利用するもの[6]，メル周波数ケプストラム係数をの特徴を利用するもの[7]，周波数帯域間のエネルギー差の特徴を利用するもの[8]などが研究されている．これらの中で，文献[8]の周波数帯域間のエネルギー差の時間遷移を利用するアルゴリズムは，MP3 などの圧縮に対して非常に頑健であり，生成アルゴリズムの各ステップが加減算などの簡単な要素で構成され，回路量が増加する乗除算が比較的少ないこと，また生成された電子指紋がコンパクトで FPGA 実装に適合しやすい利点がある．このため，このアルゴリズムは既にハードウェア実装され，ソフトウェアのおよそ 5 倍の検出速度を達成した[18]．本研究では，この実装を更にハードウェア実装向けに発展させたので，次節でこのアルゴリズムについて詳細に述べる．

2-3 Haitsma- Klker アルゴリズム[8]

図 3 に Haitsma らが提唱した電子指紋の生成ブロックダイヤグラムを示す．入力は 44.1KHz サンプリングのオーディオ信号である．これを 16K サンプル(約 370ms)毎のフレームに分割する．各フレームは 31/32 ずつオーバーラップするように取得される．別の言い方をすれば，各フレームのサンプリング開始点は 512 サンプル(11.6ms)ずつ後退しながら始まることになる．大きいオーバーラップのため，サブ電子指紋の系列は大きな類似性を持ち，時間内で少しずつ異なって行く．電子指紋は各フレーム毎に 32bit から成るサブ電子指紋が計算され，256 フレーム(256 サブ電子指紋)を結合して，8Kbit の電子指紋が生成される．

入力された 16384 ポイントから成るフレームは，FFT によって周波数領域に転写される．FFT 出力の 16384 バンドの内，300Hz から 2KHz までのバンドを等比間隔で 33 バンド選び出す．このバンド内は人間の聴覚が高い感度で動作する領域である．選び出した 33 バンドについて，エネルギーを計算する．Haitsma と Kalker は，エネルギー差は楽曲の特徴を多く含み，この特性を利用する方法は圧縮-伸張など多くの処理に頑強であることを示した[8]．

計算されたフレーム n，周波数帯 m のエネルギーを$E(n,m)$で表す．これらのエネルギー差分$ED(n,m)$は次の式によって計算される．

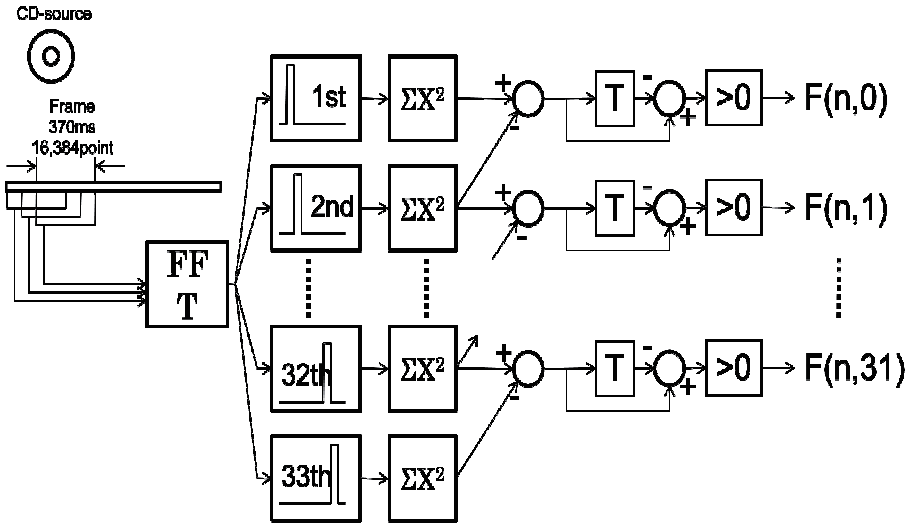

図 3 Haitsma らの電子指紋計算アルゴリズムの概要

$$\mathrm{ED}(n,m) = (\mathrm{E}(n,m) - \mathrm{E}(n,m+1)) - (\mathrm{E}(n-1,m) - \mathrm{E}(n-1,m+1))$$

サブ電子指紋のビット$\mathrm{F}(n,m)$は，次の式で算出される．

$$\mathrm{F}(n,m) = \begin{cases} 1 & \text{if } \mathrm{ED}(n,m) > 0 \\ 0 & \text{if } \mathrm{ED}(n,m) \leq 0 \end{cases}$$

ここで$\mathrm{F}(n,m)$はnフレーム目の第mビットを意味する．Haitsma らのアルゴリズムでは，バンドの差が 32 あるので，一つのサブ電子指紋が 32bit，これが 256 フレームあるので，一つの電子指紋は $32 \times 256 = 8{,}192$bit で表される．図 1，図 2 に示した電子指紋の例は，3bit×9 フレームから構成されており，1 に対応する所が黒，0 に対応する箇所が白として表現されている．音楽のデジタルデータを元に一つの電子指紋が計算されるので，入力となる音楽と電子指紋の間にはほぼ一対一の関係がある．

2-4 楽曲の同一性判定

得られた電子指紋において，任意の 2 つの全く異なる楽曲では通常全く異なる電子指紋となるが，同じ曲でも片方が MP3 への圧縮-伸張過程を経た場合など質が劣化したものである場合，それらの電子指紋はオリジナルのものと完全一致するとは限らず，若干の誤差が出てくる．そのような加工されたオーディオファイルでも識別できることはロバスト性が向上するという点で大変有用であり，ある程度の誤差も許容しなければならない．

そこで，識別を決定するパラメータとして，BER を計算する．ここでは，楽曲 A から生成された電子指紋を$\mathrm{F_A}(n,m)$，別の楽曲 B から生成された電子指紋を$\mathrm{F_B}(n,m)$とする。BER は比較する二つの電子指紋$\mathrm{F_A}(n,m)$と$\mathrm{F_B}(n,m)$のハミング距離$\mathrm{F_{diff}}(n,m) = \mathrm{F_A}(n,m)$ xor $\mathrm{F_B}(n,m)$から算出される．従って，ハミング距離に基づいた BER は，次式で計算される．

$$\mathrm{BER} = \frac{1}{32N}\sum_{n=0}^{N-1}\sum_{m=0}^{31}\mathrm{F_{diff}}(n,m) = \frac{1}{32N}\sum_{n=0}^{N-1}\sum_{m=0}^{31}\mathrm{F_A}(n,m) \text{ xor } \mathrm{F_B}(n,m)$$

ここでNはフレーム数であり，Haitsma らのアルゴリズムでは 256 となる．この二つの電子指紋のハミング距離は，楽曲 A と楽曲 B の知覚・感性的な差$\mathrm{d_\pi}(\text{Song A}, \text{Song B}')$と関係するとされている．つまり，BER の差は 2 つの楽曲間の感性の違いを数値化したものであり，2 者間で BER が低い場合，類似度が高く，ソースが同じ楽曲であるという確率が非常に高くなる．もし楽曲 B が楽曲 A を MP3 圧縮-伸張して得られた音楽ファイルであれば，BER は 0 に近い値となる．一方，楽曲 B が楽曲 A と無関係な別の楽曲である場合，BER は 0.5 に近い値となる．

```
wavedec( x[], n ){
    for( i = 0; i < n/2; i++ ){
        Wa[i] = (x[i*2]+x[i*2+1])/sqrt(2);
        Wd[i] = (x[i*2]-x[i*2+1])/sqrt(2);
    }
    return( Wa[], Wd[] );
}


band_divide( wave[], n ){
    for( i = 0; i < log2(n) - log2(32) - 1; i++ ){
        wavedec( wave, n/2^i );
        wave[] = Wa[];
    }
    wavedec( wave, n/2^8 );
    wave[] = Wd[];       /*  32 items  */
}
```

図 4 Haar 基底の Wavelet 変換を用いたバンド分割

2 個のオーディオデータ 256 フレーム分における電子指紋のブロック間のハミング距離(すなわちビットエラー)の平均を算出し,閾値 T を下回る場合,2 個の音楽信号は非常に似ているものだと判断される.この閾値 T は誤識別率P_fを決定する．つまり，音楽信号の信頼性が低いと判断されるものは，T は小さい値であり，確率P_fも小さくなる.

2-5 Haitsma- Klker アルゴリズムのハードウェア実装に対する分析

図 3 に示した Haitsma-Klker アルゴリズムをハードウェア実装の観点から分析する．第一段はフレーム分割であり，これは主にレジスタから構成される．制御部で演算は必要だが，データの操作に演算は必要としない．第二段は 16,384 要素の FFT(高速フーリエ変換)であり，これを計算するために$16384 \times 2 \times 14 \approx 460$k回の乗加算が必要となる．エネルギー計算は自乗演算なので，33 回の乗算，バンド間差分のために 32 回の減算，時間差分のために 32 回の減算が必要となる．この過程から，Haitsma らのアルゴリズムは FFT の負荷が非常に高いことが分かる．また，FFT で必要な演算は乗加算であり，乗算器は加算機に比べて FPGA 実装した場合の時間・面積負荷がおよそ 36 倍程度である．このため，FFT を用いない電子指紋計算アルゴリズムができれば，ハードウェア実装に対する適合性が大きく向上することが予想される.

3 ハードウェア実装向けオーディオ電子指紋アルゴリズム

3-1 Haar 基底の Wavelet 変換

Wavelet 変換は FFT と比較的類似の特性を持ち，Haar 基底のアルゴリズムは演算が容易であるという利点がある[14,15]．そこで本研究では，Haar 基底の Wavelet 変換を FFT の代わりに用いることによって，Haitsma らのアルゴリズムで問題となった乗加算を大幅に減らすことを試みる.

図 4 に Haar 基底の Wavelet 変換を用いたバンド分割のアルゴリズムを示す.ここで,wavedec()は Haar 基底の Wavelet 変換を行う関数,band_divide()はバンド分割を行うアルゴリズムである.関数 wavedec()は，入力として信号の時系列 x[]と配列の長さ n を受け取る．出力は，Wavelet 変換の Abstract 部分と Detail 部分になる．関数 band_divide()は，入力信号 wave[]を 8 回 Abstract 部分で Wavelet 変換し，最後に Detail 部分で Wavelet 変換する.

Haar 基底の Wavelet は次の式で計算できる.

$$W_a(i) = \{X(2i) + X(2i+1)\}/\sqrt{2}$$
$$W_d(i) = \{X(2i) - X(2i+1)\}/\sqrt{2}$$

```
Wavedec_hw( x[], n ){
    for( i = 0; i < n/2; i++ ){
        Wa[i] = x[i*2]+x[i*2+1];
        Wd[i] = x[i*2]-x[i*2+1];
    }
    return( Wa[], Wd[] );
}



                band_divide( wave[], n ){
    for( i = 0; i < log2(n) – log2(32) – 1; i++ ){
        wavedec( wave, n/2^i );
        wave[] = Wa[];
    }
    wavedec( wave, n/2^8 );
    wave[] = Wd[] / (4*sqrt(2)); /*  32 items  */
}
```

図 5 改良された Haar 基底 Wavelet 変換バンド分割

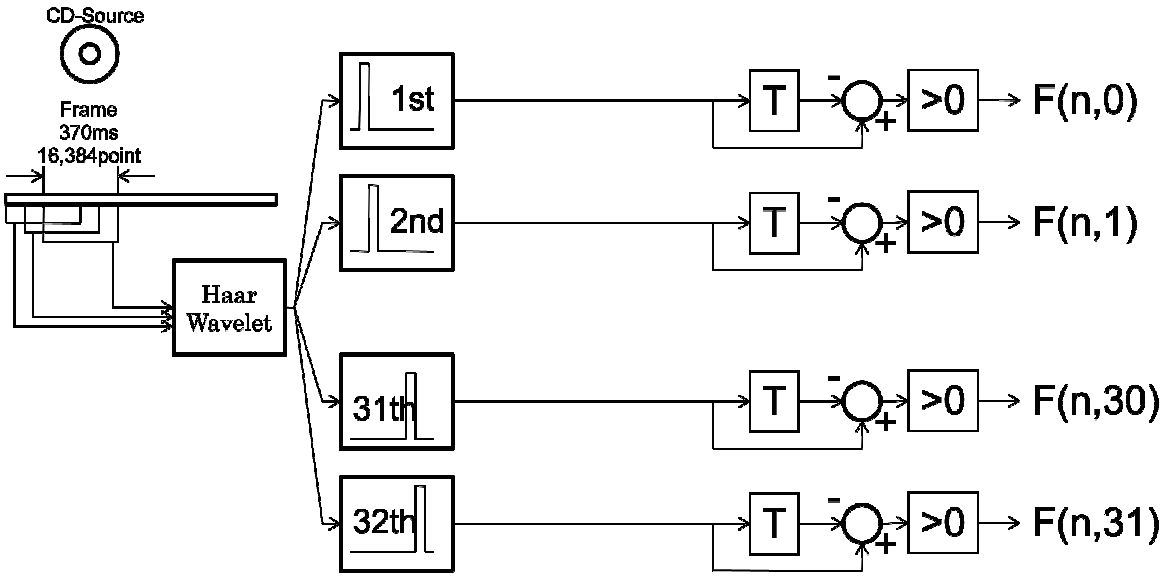

図 6　Haar 基底の Wavelet 変換を用いた電子指紋生成アルゴリズム

ここでX(i)は入力信号の時系列である．Wavelet 変換を行うと，信号の時系列は半分になる．Haar 基底の Wavelet 変換では，Abstract 部分(Wa)と Detail 部分(Wd)に分かれる．1 フレーム当りのサンプル数は 16,384，後段で必要とされるバンド数は 32 で，一回の Wavelet 変換で信号の長さは半分になるため，$\log_2 16384 - \log_2 32 = 9$ 回の Haar 基底 Wavelet 変換を行えば良い．Haitsma らのアルゴリズムでは，バンド間の差分を計算していた．Wavelet 変換の Detail 部分の計算は，隣接する信号の差分であり，FFT における差分計算に相当すると考えられる.そこで,提案するアルゴリズムでは,8 回の Haar 基底 Wavelet 変換の Abstract 部分を取得し，最後の Wavelet 変換で Detail 部分を取得する．

入力信号は16bit×16,384サンプルである.図4のアルゴリズムから分かるように,Haar基底のWavelet計算は殆どが加減算のみである．図 4 では 2 の平方根での除算が含まれているが，これは繰り返し演算の中ではなく，最後に一回演算するだけで済ますことができる．この 2 の平方根による除算を削減したアルゴリズムを図 5 に示す.

Symbol	Song	Re-sampling Software	Modification
PCM(16bit-44.1kHz)	Song1～10	(Original)	
MP3	Song1'～10'	LAME	CBR128k-44.1kHz
WMA	Song1"～10"	WMA9	CBR128k-44.1kHz
Sampling rate change	Song1'"～10'"	Lilith	22.05kHz

表 1 実験に用いた楽曲の加工

3-2 ハードウェア実装向け電子指紋計算アルゴリズム

図 6 に Haar 基底の Wavelet 変換を用いた電子指紋生成アルゴリズムのブロックダイヤグラムを示す. 図 3 の Haitsma らのアルゴリズムと比べて，FFT の部分が Haar 基底 Wavelet 変換になっていることが本アルゴリズムの特徴である．FFT の演算は，$16384 \times 14 \approx 230$k回の乗加算が必要であったが，図 5 の Wavelet 変換では $16383 \times 2 \approx 32$k回の加減算で済むことが分かる．一般に FPGA 上では乗算器は加算器の 10 数倍程度の面責を必要とするので，FFT 部分は Wavelet 変換にすることによって，およそ 100 分の 1 に削減できる．

Wavelet 変換後の回路は Haitsma らのアルゴリズムとほぼ同じである．先に述べたように，Wavelet 変換の最終段を Detail 部分にすることによって，バンド間の差分を得るのと同等の効果を得る．この後，時間差分を経て，電子指紋 $F(n,m)$を得る．楽曲の同一性の判定は，2-4 節の方法と同様である．

4 性能評価

4-1 圧縮-伸長に対する耐性，および誤識別率

（1）音楽ファイルの加工

提案アルゴリズムの圧縮-伸長に対する耐性を評価するために,10 曲の音楽に対して 3 通りの加工を行った.この一覧を表 1 に示す.オリジナル音楽ファイルは 16bit 44.1KHz サンプリングの PCM 方式(Windows wave フォーマット)である．MP3 は，音楽を一旦 CBR128kbps の MP3 ファイルに圧縮した後，44.1KH z サンプリングに伸長する加工,WMA は,同様に音楽を一旦 CBR128kbps の WMA ファイルに圧縮した後,44.1KH z サンプリングに伸長する加工，また Samling rate change は，オリジナル音楽ファイルのサンプリングレートを一旦 22.1KHz サンプリングに落とし，それを補完して元の 44.1KHz サンプリングに戻すという加工を行った．これらの加工を Song 1～Song 10 までの 10 曲について行った.

（2）圧縮方式ごとの BER

表 2 に提案アルゴリズムによる電子指紋の組み合わせによる BER を示す.表中,例えば Song1 と Song1'は，オリジナルの音楽ファイルから生成された電子指紋と，オリジナルの音楽ファイルを MP3 圧縮し伸長したファイルを入力として生成された電子指紋の BER を示している.10 曲について評価したので,Song 1～Song10 までの項目があり，加工方式が 3 通りあるため，Song1～Song1''' まで項目がある.

同じ音楽ファイルを入力とする場合(例えば Song1'と Song1')は，入力が同じ音楽ファイルなので,生成される電子指紋も完全に一致し，従って BER は 0 となる． 楽曲が同じで加工方式が異なる場合(例えば Song1 と Song1'，Song1''と Song1'''の組み合わせ)，入力となる音楽ファイルはほぼ同じだが,非可逆圧縮を経ているため，完全に一致するわけではない．従って，生成される電子指紋もほぼ一致するが,完全には一致しない.そこで,BER は 0 に近い数値となる.楽曲が異なる場合(例えば Song1 と Song10)や，楽曲も加工方式も異なる場合(Song1'と Song10'')には，入力となる音楽ファイルが異なるので,生成される電子指紋も全く異なる．従って BER は 0.5 に近い数値となる.

電子指紋のアルゴリズムの優劣は次のように評価することができる．元の入力が同じ音楽ファイルである場合，加工の有無や方式の違いに関わらず，BER が 0%に近い数値となり，また元の入力ファイルが

BER (%)		Song1				…	Song10		
		Song 1	Song 1’	Song 1”	Song 1”’			Song 10”	Song 10”’
Song1	Song1	0	3.5	5.8	1.2			48.2	51.6
	Song1’		0	2.2	4.2			50.5	49.4
	Song1”			0	1.8			51.3	52.2
	Song1”’				0			49.8	50.3
	・・・								
Song 10	・・・								
	Song10”							0	
	Song10”’								0

表 2 電子指紋の BER

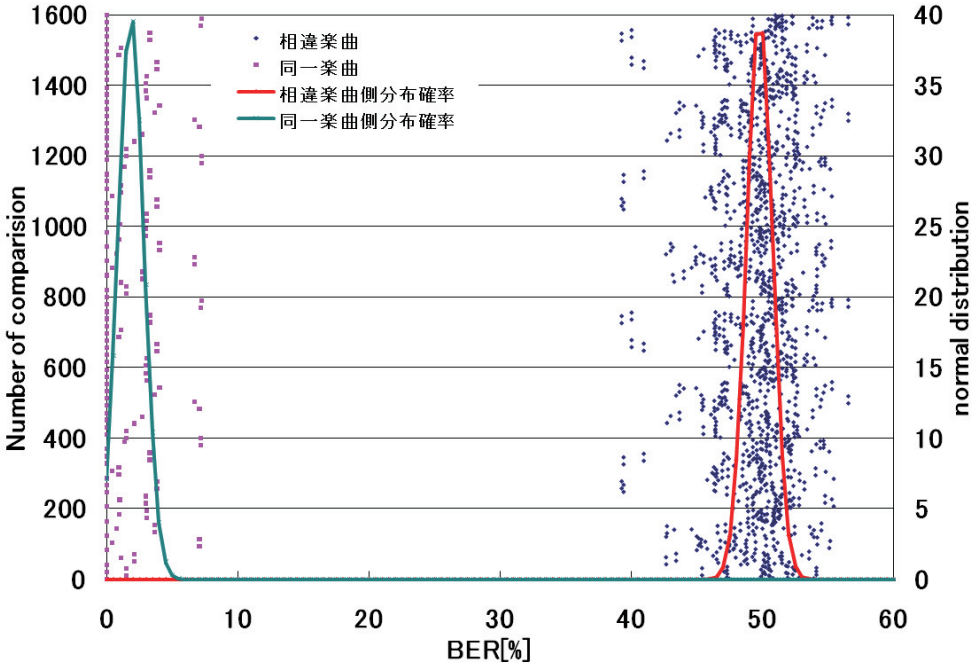

図 7　提案アルゴリズムの分解能

異なる場合，加工の有無や方式の違いに関わらず，BER が 50%に近い数値となるアルゴリズムが良いアルゴリズムであると言える．楽曲の同一性判定には，BER がスレッショルド Th より低ければ同じ楽曲と判定し，BER が Th よりも大きければ違う楽曲と判定する．同じ楽曲にも関わらず BER が高い数値となったり，異なる楽曲なのに BER が小さい数値になると，誤識別になる．そこで，次節では誤識別率について議論する．

（３）提案アルゴリズムの識別精度

表 2 の全体について，密度分布のグラフにしたものを図 7 に示す．図中縦軸は組み合わせの番号であ

	FFT Based (Radix 2)	Wavelet Based	Ratio
# of R. Adder	16,975,141	16,416	1,034
# of FFs	458,881	16,416	27.95
Processing time	274 clock	267 clocks	1.026

表 3 演算器数

る．四角点は元の音楽ファイルが同じもの同士の組み合わせを示し，ダイヤ点は元の音楽ファイルが異なる組み合わせを示す．理想的には四角点が 0%に集中し，ダイヤ点が 50%に集中するグラフとなるはずである．図の実線は，点の数を正規分布に従うと仮定した場合の正規分布曲線である．

同一楽曲同士の BER は，平均 1.860%で標準偏差 2.207，また異なる楽曲同士の BER は，平均 49.754%で標準偏差 2.913 となった．これらが正規分布であると仮定すれば，両分布曲線の交点がスレッショルドになるべきである．計算したところ，交点は BER=23%となった．つまり，スレッショルド Th=23%となり，2 つの入力による電子指紋の BER を計算した時に，**BER < Th**ならば同一楽曲，**BER > Th**ならば違う楽曲と判定できることになる．

一方，正規分布曲線は BER=Th の所でもゼロにはならず，非常に小さい値を取る．異なる楽曲同士の BER が Th 以下となる確率，また同じ楽曲にもかかわらず BER>Th となる確率が誤判定率と考えることができる．正規分布曲線に従うと仮定すれば，平均 1.860%，標準偏差 2.207 の正規分布が BER>23%になる確率，および平均 49.754%，標準偏差 2.913 の正規分布が BER<23%になる確率が誤判定率であり，10^{-30}以下となることが分かった．

4-2 ハードウェアへの実装性

（１）演算器数

表 3 に，Haitsma らのアルゴリズム(表中”FFT Based”)，および提案アルゴリズム(表中”Wavelet Based”)をハードウェア化する場合に必要な回路量を示す．表中”Ratio”は，両者の比率である．

Haitsma らのアルゴリズム(FFT Based)による FFT では，これまでの実装経験より，実数乗加算器 1 つあたり加算器の約 37 倍の回路量が必要となることが分かっている．16,384 ポイントの FFT なので，**16384 × 2 × 14**回の乗加算が必要となるため，加算器換算として約 17 百万個の加算器に相当する回路量が必要となる．また，エネルギー計算やバンド間差分，時間差分の計算のためには，加算器換算としてたかだか加算器 1,300 個程度の回路量が必要となるである．

一方，提案アルゴリズム(Wavelet Based)では，Wavelet 変換に大きな回路量が必要であるが，この計算は固定小数点の加減算のみで処理できるため，乗算器を含まず，約 16 千個の加算器が必要となる．後段の処理は Haitsma らのアルゴリズムと殆ど同じである．

同様にフリップフロップ(FF)や処理時間について所用回路量および所用クロック数について見積もった結果を表 3 に示した．

（２）処理能力

表 4 に，Haitsma らのアルゴリズム(表中”FFT Based”)，および提案アルゴリズム(表中”Wavelet Based”)をハードウェア化した場合の推定処理能力を示す．Haitsma らのアルゴリズムの FPGA 上への実装は，文献[18]によって行われており，”FFT Based” の処理能力についてはこれを引用した．実験プラットホームとして CPU は 2.8GHz PentiumIV プロセッサ，FPGA デバイスは Xilinx 社 XC2V6000 を用いている．図 8 に実験システムの外観，図 9 に実験システムで用いた FPGA ボードを示す．

表 4 より，Haitsma らのアルゴリズムを FPGA によってハードウェア化した場合，ソフトウェア処理に比べておよそ 5 倍～10 倍の速度向上率が得られていることが分かる．提案アルゴリズムは FFT に代わり Haar 基底の Wavelet 変換を用いるが，この処理ステップ自体ソフトウェアでも十分高速化が可能であり，ソフトウェア実装された提案アルゴリズムは，Haitsma らのアルゴリズムのソフトウェア実装よりもおよ

		Time for FP Calculation	Search Time	Performance Ratio
FFT based	Software	3,353ms (19.5Mbps)	-	1.0
	Hardware with Radix-2 FFT	650.24ms (102Mbps)	0.543ms	5.152
	Hardware with Radix-4 FFT	314.62ms (213.3Mbps)	0.543ms	10.66
Wavelet Based	Software (Examined)	128ms (524.3Mbps)	-	26.89 1.0
	Hardware (Estimated)	0.623ms (650.24ms/1034) (105.5Gbps)		5,382 205.5

表 4 推定処理能力

図 8 実験システムの外観

図 9 実験に用いた FPGA ボード (Celoxica 社 RC2000)

そ 26 倍程度処理速度が高速である.更にハードウェア実装された Haitsma らのアルゴリズムに対しても，2.5 倍程度高速であり，アルゴリズムの改良による速度向上の効果が大きいことが分かる．ハードウェア実装した場合，表 3 より，提案アルゴリズムの所用回路量は Haitsma らのアルゴリズムに比べておよそ 1/1,000 倍と大きく縮小できる．逆の言い方をすれば，同じ面積の LSI チップが与えられた場合，提案アルゴリズムは 1,000 倍以上の並列化が可能である．表 3 より，所用クロック数は提案アルゴリズムも Haitsma らの方法もほぼ同じであることが分かり，またインターネット中を通過する音楽ファイルの電子指紋検出においては，並列化がほぼ線形にスループットの向上につながる．この特性を考慮し，ハードウェア実装された提案アルゴリズムのスループットを推定すると，およそ 105Gbps となり，インターネットでの音楽ファイルの処理には十分な処理能力を実現できることが分かる．更にソフトウェア実装からハードウェア実装への転換で，Haitsma らのアルゴリズムは 10 倍程度の速度向上になっているのに対し，提案アルゴリズムでは 205 倍の速度向上率となっている．提案アルゴリズムはハードウェア実装に対する適合性が非常に高い．

5 まとめと今後の課題

本研究では，近年増加するインターネットを介した音楽ファイルの流通制御を目的として，大きな処理能力向上が期待できるハードウェア向けの電子指紋検出アルゴリズムの提案とハードウェア実装にたいする適合性の評価を行った．提案アルゴリズムは，従来 FFT でバンド分割していた所を，ハードウェア実装した場合に大変簡素な回路で構成できる Haar 基底の Wavelet 変換に変更したところに特徴がある．

提案アルゴリズムの精度を評価するために,複数の方式で圧縮-伸張した音楽ファイルを入力として判定精度を評価したところ，ソースが同じで圧縮-伸張方法が異なる音楽ファイル同士は十分小さい値を出力し，ソースが異なる音楽ファイル同士の場合は，ほぼ BER が 0.5 近辺になることが分かった．また，その標準偏差は十分小さく，分離が明確にできることも分かった．正規分布に従うと仮定した場合の誤識別率は，10^{-30} 以下であることも明確になった．

提案アルゴリズムのハードウェア実装への適合性を評価したところ，従来手法に比べて使用演算器数が約 1/1000 以下，またハードウェア実装した場合のスループットは 105Gbps が期待できることが分かった．

今後の課題として，インターネットに適用するための周辺システムの構築，誤識別率をさらに低下させるような付加処理の開発，また，FPGA への実装にあたって実現可能な，実装レベルの最適化があげられる．

謝辞

本研究の遂行にあたって,南フロリダ大学への派遣旅費を助成して頂いた電気通信普及財団 長期海外渡航研究制度および関係各位に深く感謝いたします．

【参考文献】

[1] Pedro Cano, Elio Batlle, Ton Kalker and Jaap Haitsma, “A Review of Audio Fingerprinting”, Journal of VLSI Signal Processing, Vol. 41, pp. 271-284, 2005

[2]Miller, M.L., Rodriguez, M.A. and Cox, I.J. “Audio Fingerprinting: Nearest Neighbor Search in High Dimensional”, IEEE Workshop on Multimedia Signal Processing, pp. 182-185, 2002

[3] P. Cano, E. Batlle, H. Mayer, and H. Neuschmied, “Robust sound modeling for song detection in broadcast audio,” in AES 112th International Convention, May 2002.

[4] E. Allamanche, J. Herre, O. Hellmuth, B. Frbach, and M. Cremer, “Audioid: Towards content-based identification of audio material,” in 100th AES Convention, May 2001.

[5] A. Wang, “An industrial strength audio search algorithm,” in 4th Int. Symposium on Music Information Retrieval (ISMIR), Oct.2003

[6] Y. Cheng, “Music database retrieval based on spectral similarity,” in 2nd Int. Symposium on Music Information Retrieval (IS-MIR), Oct. 2001.

[7] P. Cano, E. Batlle, T. Kalker, and J. Haitsma, “A review of algorithms for audio fingerprinting”, in International Workshop on Mutimedia Signal Processing, Dec. 2002.

[8] Jaap Haitsma , Ton Klker “A Highly Robust Audio Fingerprinting System” Proc. ISMIR 2002 3rd International Conference on Music Information Retrieval

[9] Sert, M., Baykal, B. and Yazici, A., ”A Robust and Time-Efficient Fingerprinting Model for Musical Audio”, IEEE International Symposium on Consumer Electronics, pp. 1-6, 2006

[10]Shumeet Baluja and Michele Covell, “Content Fingerprinting Using Wavelets”, 3rd European Conference on Visual Media Production, pp. 198-207, 2006

[11] Felix Balado, Neil J. Hurley, Elizabeth P. McCarthy and Guenole C.M. Silvestre, “Performance of Philips Audio Fingerprinting Under Additive Noise”, IEEE International Conference on Acoustic, Speech, and Signal Processing, Vol. II, pp. 209-212, 2007

[12] Kazuhiro SAKAKIBARA,Yasushi INOGUCHI, "Detection of the audio watermark on FPGA", CPSY2004-80, VLD2004-114, pp. 25-30

[13] Prarthana Shrestha, Ton Kalker "AUDIO FINGERPRINTING IN PEER-TO-PEER NETWORKS", 2004 Universitat Pompeu Fabra.

[14] N. Ahmed, T.Natarajan, and K.R. Rao, “Cooley-Tukey-Type Algorithm for the Haar Transform”, Electronics Letters, Vol. 9, No. 12, pp. 276-277, 14th June 1973

[15] Yi Chu, Wen-Hsien Fang and Shun-Heyung Chang, “n Efficient Haar Wavelet-based Approach for the Harmonic Retrievalproblem”, IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3, pp. 1969-1972, 1997

[16] 小野 束, 電子透かしとコンテンツ保護, オーム社, 2001.

[17] 安田 浩, 安原 隆一,コンテンツ流通, アスキー, 2003.

[18] 磯永 久史, 井口 寧 , “FPGA を用いた高速オーディオフィンガープリントシステムの構築”, 信学技法 RECONF2005-77, Vol. 105, No. 513, pp. 1-6, Jan. 2006.

〈発 表 資 料〉

題　　名	掲載誌・学会名等	発表年月
High speed audio fingerprint detection algorithm for hardware implementation	Seminar in Center for Communications and Signal Processing and Center for Digital and Computational Video, University of South Florida, ENB261	2009 年 3 月
TTN: A High Performance Hierarchical Interconnection Network for Massively Parallel Computers	IEICE Transactions on Information and Systems, Vol. E92-D, No. 5, pp.1062-1078	2009 年 5 月
Improving accuracy of host load predictions on computational grids by artificial neural networks.	In 11th Workshop on Advances in Parallel and Distributed Computational Models held in conjection with 23rd IEEE International Parallel and Distributed Processing Symposium., 8 pages in CD-ROM, Rome, Italy	2009 年 5 月
Dynamic communication performance of a TESH network under the nonuniform traffic patterns	In 11th International Conference on Computer and Information Technology (ICCIT 2008), pp.365-370, Khulna, Bangladesh	2008 年 12 月
On hot-spot traffic pattern of TESH network	In 11th International Conference on Computer and Information Technology (ICCIT 2008), pp.359-364, Khulna, Bangladesh	2008 年 12 月
Multimedia and Learner Awareness - Raising in Regard to Japanese Prosody	The Third CLS International Conference 2008, pp.481-486, NUS, Singapore,	2008 年 12 月
動的再構成デバイスにおけるデータ I/O と処理のオーバーラップを 用いた処理法	先進的計算基盤システムシンポジウム SACSIS, pp.55-56	2008 年 6 月
マルチコンテキスト型リコンフィギャラブルプロセッサに おけるデータ並列タスクの処理法	信学技法 RECONF2008-41, Vol. 108, No. 300, pp.15-20	2008 年 11 月
マルチコアクラスタに対するマルチコアプロセッサを意識した 並列処理手法に関する研究	2008 年度 電気関連学会 北陸支部連合大会, E-62	2008 年 9 月