I-005

抽出ピッチの拡張による PICOLA アルゴリズムの高速化

Acceleration of PICOLA Algorithm by Multiplying Extracted Pitches

清水 省吾† 藤澤 義範‡ 伊藤 祥一‡
Shogo Shimizu Yoshinori Fujisawa Shoichi Ito

1．はじめに

近年，高齢者や障害者のためのユーザビリティとして，アクセシビリティが提唱されている．映像や音声といったコンテンツにおけるアクセシビリティの一つとして，コンテンツの再生速度を変更する低速再生がある．例えばフリーの映像再生ソフトである VLC Media Player[1]では，動画再生時に再生速度の変更を行うことができる．また，Apple は開発者向けにフレームワークとして AVAudioPlayer[2]というライブラリを用意し，0.5～2.0 倍速までの音声再生速度が変更できるようになっている．

一方で音声の再生速度の変更において，音声を単純に時間軸へ伸長および圧縮すると，音声のピッチが変化してしまう．ピッチが変化することで，元の音声の印象が変化してしまうため，ピッチが変化しない時間軸方向の伸長圧縮が必要である．これをタイムストレッチという．これは前述の VLC Media Player や AVAudioPlayer の時間軸方向の伸長圧縮処理にも採用されている．タイムストレッチには，いくつかの方法が存在するが，中でも森田直孝，板倉文忠らが提唱した「ポインタ移動量制御による重複加算法 – PICOLA（Pointer Interval Controlled OverLap and Add）[3]」と呼ばれる手法には定評がある．これは自己相関法によりピッチの抽出を行い，ピッチごとに処理を行うため，高速にタイムストレッチを行うことができる．PICOLA による伸長圧縮アルゴリズムは，ソニー株式会社[4]や株式会社リコー[5]といった企業の製品にも採用されている．

しかし，比較的高速である PICOLA であっても，ストリーミング配信のようなサービスで随時クライアント側によるタイムストレッチを行おうとすると，特に高速再生において変換処理が間に合わなくなるという問題がある．近年ではスマートフォンやタブレットといったマシンパワーの小さい端末でのコンテンツ利用が増えたため，タイムストレッチ手法の高速化が期待されている．そこで筆者らは PICOLA の自己相関法によって得られた抽出ピッチ幅を整数倍に拡張することで，伸長圧縮処理の高速化を行い，高速化率や特性に関する検証を行った．

2．PICOLA における伸長圧縮手法

本章では PICOLA による伸長圧縮の方法について述べる．

2.1 ピッチの抽出

PICOLA では，1 波長ごとに伸長圧縮の処理を行うため，処理を行うピッチを抽出する必要がある．そこで自己相関法という手法を用いてピッチの抽出を行う．処理を行う波形のモデルを図 1 に示す．

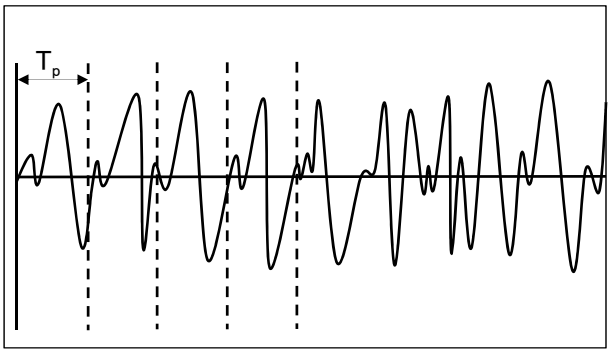

図 1：自己相関法のため音声波形モデル

自己相関法は自分と同じ形の波形が次にどこに出てくるのかを，探索窓の大きさを変えながらピッチ抽出を行う方法である．長さ T_p の窓を用いる場合，探索位置から T_p だけシフトした箇所との差分を計算し，探索開始位置から T_p-1 地点までの差分の合計から差分平均値の計算を行う．これを窓の大きさを変化させながら，あらゆる大きさの窓の差分平均を比較し，最も差分が小さい窓の大きさが抽出ピッチとなる．図 1 の点線で示された間隔が，自己相関法で得られたピッチ幅である．ピッチ抽出では自己相関法以外に AMDF や−1, 1 パターン相関法が存在するが，これらのどの手法を使用してもタイムストレッチ処理への影響は無いことが知られている[6]．

しかし自己相関法によるピッチ抽出では，探索窓が音源のサンプル数に比べ極端に小さすぎると，正しくピッチ抽出ができないという問題がある．そこで探索窓の大きさを人間の可聴域である 20Hz～20kHz の間にレンジさせることで，正しくピッチ抽出を行うことができる．なお，自己相関法ではこれ以外の問題点として，想定したピッチではなく，倍音としてピッチが抽出されてしまうという問題もある．しかし PICOLA の処理では，次の波形とより類似性の高い部分がピッチとして抽出できた方がオーバーラップ処理の際にノイズになりにくいため，問題無く伸張および圧縮処理が行える．

2.2 PICOLA による伸長圧縮処理

はじめに自己相関法によって抽出されたピッチを T_p とし，伸長圧縮レートを R とする．例えば 3 倍の長さに波形を伸長する場合は R を 3 とし，1/3 の長さになるように波形を圧縮する場合は R を 1/3 とする．

伸長処理では，長さ L を $L = T_p / (R-1)$ と定義する．次に伸長圧縮処理を行う波形地点を A 点とし，A 点から L だけ離れた箇所を B 点とする．AB 点が定まったら，両点に対して A 点は 0 から 1，B 点は 1 から 0 となるように重み付けした長さのオーバラップ波形を生成し，長さ T_p の出力

† 長野工業高等専門学校 専攻科 電気情報システム専攻
‡ 長野工業高等専門学校 電子情報工学科

波形とする．最後に A 点から長さ L の波形を出力し，B 点を次の伸長処理の開始点である A’点とする．この処理では長さ L から T_p+L の長さになる．L の定義を用いると T_p+L は式(1)のように変形できる．つまり，T_p+L はもともとの波形の長さ L が R 倍に伸張されたことを示している．

$$\begin{aligned} T_p + L &= T_p + \frac{T_p}{R-1} \\ &= \frac{RT_p - T_p + T_p}{R-1} \\ &= R\frac{T_p}{R-1} \\ &= RL \end{aligned} \quad (1)$$

圧縮処理では，長さ L を L = RT_p / (1-R)と定義して考えると，式(2)から T_p+L を R（R<1）倍し L を得ていることから R 倍の圧縮を実現していることがわかる．

$$\begin{aligned} L &= \frac{RT_p}{1-R} \\ &= \frac{R\left(T_p + RT_p - RT_p\right)}{1-R} \\ &= R\left(\frac{RT_p}{1-R} + \frac{T_p - RT_p}{1-R}\right) \\ &= R(T_p + L) \end{aligned} \quad (2)$$

3. ピッチの拡張による PICOLA の高速化

本章ではピッチを拡張する方法による PICOLA の高速化手法について述べる．

3.1 原理

PICOLA では一度に処理できる波形の長さは，伸張と圧縮のそれぞれの L を使って表現すると，伸長の場合は L，圧縮の場合は T_p+L となっている．そのため，音声データ全体の波形の長さを T_d とし，音声データから取得可能な L の平均値を L_a とすると，図 2 のように伸長処理の全体の処理回数は T_d / L_a となる．圧縮処理では T_d / (L_a+T_p)となる．L_a は L の平均値であり，L の定義より，伸長と圧縮のどちらにおいても T_p に比例する量であるため，処理の回数は T_p に反比例することがわかる．

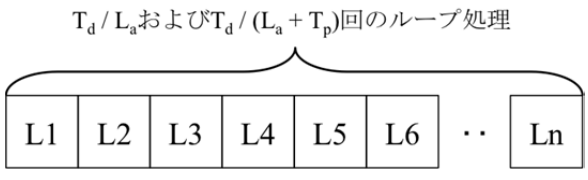

図２：伸張圧縮処理の処理回数

つまり，一度に処理を行うピッチ幅 T_p を大きくすることが，全体の処理の回数を減らすことにつながることがわかる．PICOLA では内部処理にかかる時間の半分以上がピッチ抽出のため，ピッチ抽出の回数を減らすことで全体の処理速度の高速化を実現できると考えられる．

そこで本研究では，自己相関法によって取得したピッチを整数倍に拡張することで，処理全体の繰り返し回数の削減を行った．音声は振動であり，ある程度同じ形の波形が連続して出現することで音声として認識される．つまり隣り合う 1 周期分の波形では，その型状に一定の相関が見られる．図 3 は「あー」と発話した時の音声波形の一部である．

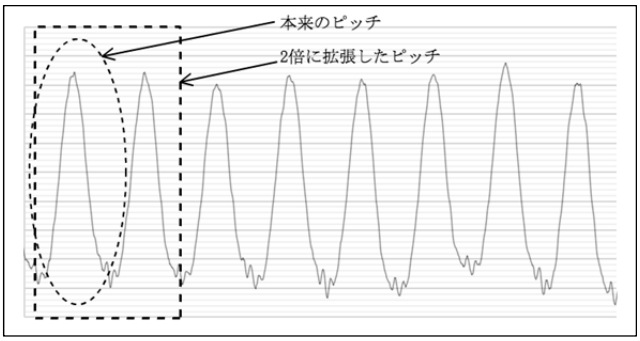

図 3：実際の発話時の波形とピッチ拡張

図 3 のように，一般的には丸枠で囲んだ部分がこの波形のピッチとなる．しかし PICOLA の伸長圧縮処理では，オーバーラップ波形を生成する際に，重ね合わせを行う波形同士が類似していればよい．そのためピッチを整数倍に拡張することで，一度に処理できる量を増やすことができる．図 3 の波形では 2 倍に拡張した波形も，次に出現する波形とほぼ同じ型状であることがわかる．このような処理を導入することにより，図 4 のように PICOLA の処理回数も減らすことができると考えた．

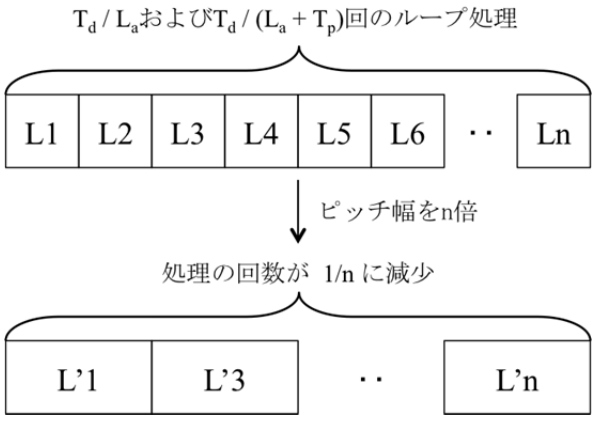

図４：ピッチ拡張した時の処理回数

3.2 高速化による PICOLA への影響

高速化手法では，長さ T_p のピッチ幅を n 倍に拡張した波形と，その次に出現する波形が類似していることが条件である．図 5 に示す波形は，実際にピッチを拡張した場合に問題が起きるケースである．図 5 では同じ波形のピッチを 2 倍および，3 倍に拡張したものを示している．2 倍に拡張したものは，次に出現する長さ $2T_p$ の波形も，類似している．しかし，3 倍に拡張したものは，次に出現する $3T_p$ の波形との類似性が弱くなっている．PICOLA では前後の波

形同士が連続になるようにオーバーラップ処理がされるため，波形の類似性が小さいまま処理を行うと，波形同士をつなぎ合わせるオーバーラップ部に両方の波の特性が出るという問題がある．例えば 2 倍にピッチ幅を拡張して 2 倍の伸張処理を行った場合，拡張したピッチの影響は出力に 4 波長分現れる．また，440Hz 音を 2 倍に伸長処理した場合，10 倍にピッチを拡張すると 20 波長分の出力に影響がおよぶ．このとき，1 波長の長さは約 2.2 ミリ秒であるため，45 ミリ秒の間，出力にエコー音が入る可能性がある．また 2 倍の圧縮では 11 ミリ秒の間，波形に影響がおよぶことが計算により求めることができる．

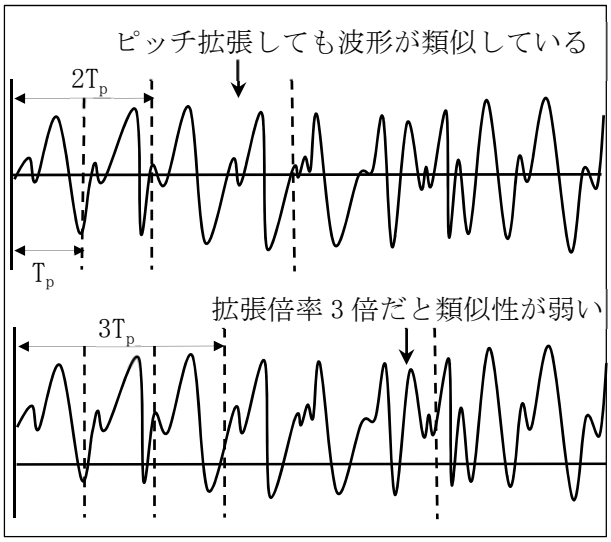

図 5：ピッチの拡張による類似性の消失

4. 高速化率の検証

本章ではピッチ拡張法による高速化率の測定について述べる．

4.1 測定方法

高速化率の測定は，2 種類の方法で行った．1 つ目はピッチの拡張倍率を変化させた時の，PICOLA の伸長圧縮に要する CPU 時間の測定である．これは WAVE 形式の音源についてピッチ倍率を変えながら伸長圧縮処理を行う．同時に伸長圧縮の倍率も変化させながら，高速化による時間短縮について検証する．また，サンプリングレートの変化による高速化率についても同時に計測する．2 つ目は音源の違いによる高速化率の検証である．これはランダムに選択された同一のデータ量かつ同一サンプリングレートを持つ音源についてピッチ倍率を変えながら伸長圧縮処理を行い，どの程度 PICOLA による伸張圧縮処理時間にばらつきがあるのかを検証する．また今回の検証は次の環境により行った．

実験環境

OS：	Mac OS X 10.10.2
プロセッサ：	2.6GHz Intel Core i5

使用したデータ

量子化ビット：	8 ビット
チャンネル数：	モノラル
音源の再生時間：	20 秒

4.2 測定 1-ピッチ拡張による高速化測定

サンプリングレート 44100Hz の楽曲音源を，ピッチ倍率を変化させながら伸長した際の CPU 時間を計測した結果を図 6 に，圧縮した結果を図 7 に示す．

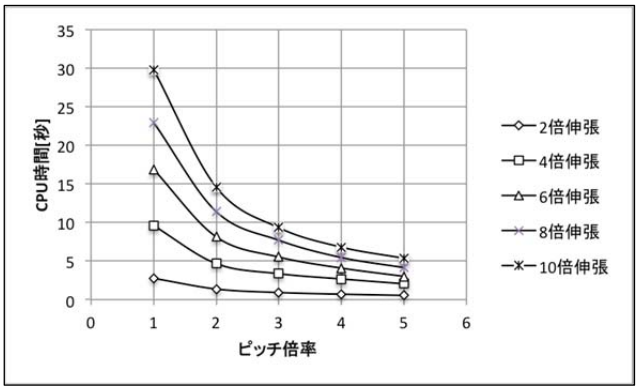

図 6：ピッチ拡張による伸張処理時間の変化

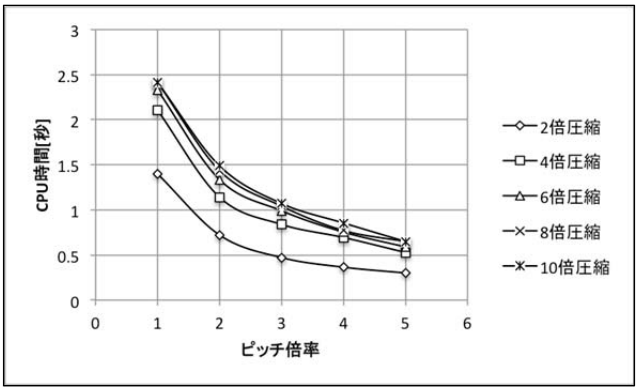

図 7：ピッチ拡張による圧縮処理時間の変化

図 6，7 では，2 倍～10 倍までの伸長および圧縮処理にかかる時間を，それぞれピッチ幅の倍率を 1 倍～5 倍に変えながら計測を行っている．結果から，ピッチ幅が増加したときに，すべての状態において，処理時間の短縮がされている事がわかる．また，ピッチ幅の増加による時間変化が反比例の関係で現れていることがわかる．

次に圧縮率を 2 倍に固定しサンプリングレートごとにピッチ幅を変化させたときの伸長処理時間を計測した結果を図 8 に，圧縮処理時間を計測した結果を図 9 に示す．

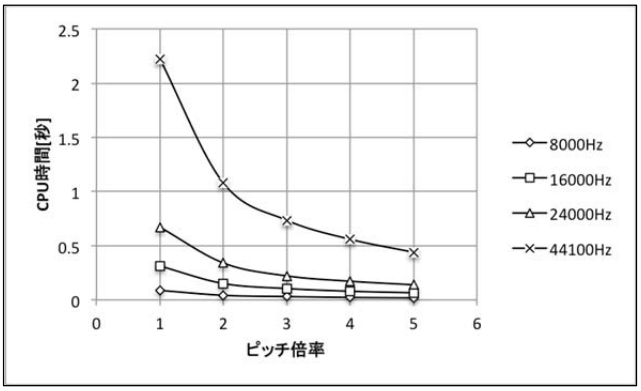

図 8：サンプリングレートごとの伸長処理時間変化

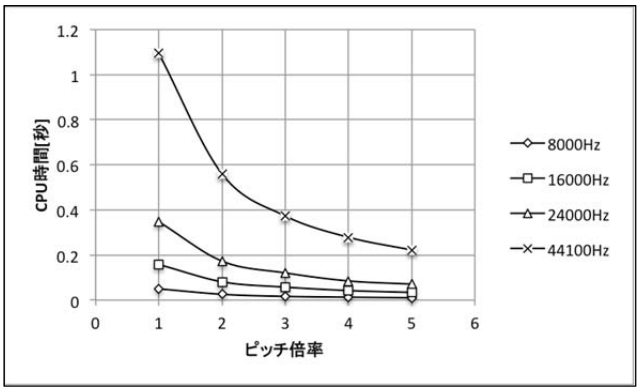

図 9：サンプリングレートごとの圧縮処理時間変化

図 8 と図 9 はそれぞれ 2 倍伸長及び 2 倍圧縮に要した時間をピッチ幅の倍率を高めながら計測し，同一音源でサンプリングレートが異なるファイルごとにグラフ化を行った．結果としては，すべての音源に対して反比例の関係による処理時間の短縮が見られた．

4.3　測定 2-音源の違いによる高速化測定

本節では様々な種類の音源に対してピッチの拡張による PICOLA の高速化測定を行い，処理時間短縮にどのような変化があるかについて検証を行う．PICOLA の処理では音源の周波数特性によって処理速度が異なる．そこでピッチの拡張が音源の違いによって，どこまで PICOLA の処理に影響するか調査を行った．音源はすべて同じデータサイズであり，次の 5 種類を選択した．

- 男性合唱曲：　男性声の低音が中心
- 女性合唱曲：　女性声の高音が中心
- ピアノ曲：　　楽器音，高温域の楽曲
- オーケストラ：様々な楽器及び男女の合唱
- 1kHz 音：　　1kHz のビープ音

それぞれの音源について，ピッチ拡張による伸長処理の高速化検証結果を図 10 に，圧縮処理の結果を図 11 に示す．

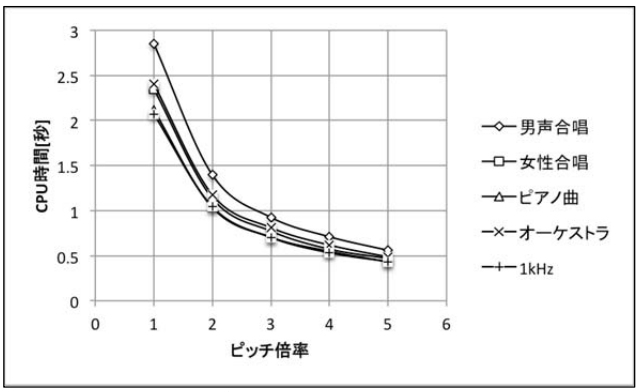

図 10：音源の違いによる伸長処理時間の変化

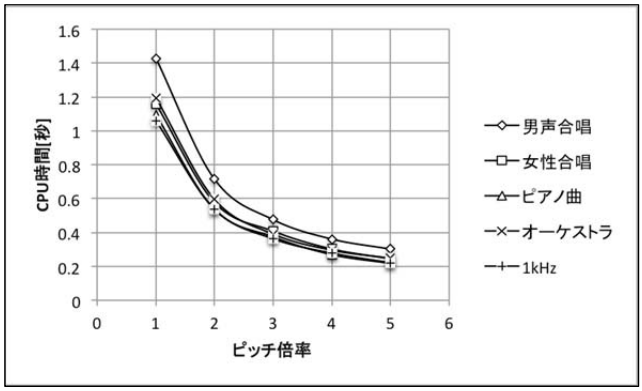

図 11：音源の違いによる圧縮処理時間の変化

図 10，11 から，伸長と圧縮ともに，各音源ごとに処理にかかる時間に変化はあったものの，ピッチ幅を変えた時の時間短縮の変化量は，一定であった．

5．まとめと考察

本論文ではピッチ幅の拡張による，PICOLA の高速化について述べた．本提案手法に基づいて PICOLA アルゴリズムを実装したアプリケーションを筆者らはすでに開発している．アプリケーションの使用者からは，旧来に比べて処理が高速になったが，品質の劣化はほとんどなかったといった感想を得ている．本手法ではピッチ幅に比例して，処理速度が向上していることがわかった．また処理速度の向上の度合いは，音源の伸長圧縮率やサンプリングレート，音源の種類のすべての変化によって影響されないことがわかった．本手法を採用することで，PICOLA 処理の大幅な高速化を実現することができた．しかし，3.2 節で述べたように，ピッチ幅と伸張圧縮率の積が一音の発話時間に対して長くなると，エコーのような音が生成されるという問題がある．今後は本手法による時間短縮を行った音源が，どれほどのピッチ幅なら人間が聞き取りにくいのかという検証や，音質の定量評価による，品質が劣化しやすい音源の種別を行う必要がある．

参考文献

[1] VideoLAN，VLC: オフィシャルサイト，http://www.videolan.org/index.ja.html
[2] Mac Developer Library，AVAudioPlayer Class Reference，https://developer.apple.com/library/prerelease/mac/documentation/AVFoundation/Reference/AVAudioPlayerClassReference/index.html
[3] 森田，板倉，“ポインター移動量制御による重複加算法（PICOLA）を用いた音声の時間軸での伸長圧縮とその評価”，日本音響学会講演論文集，pp.149-150，1986.
[4] ソニー株式会社，【特開 2001-125600】再生速度変換装置及び方法，http://www8.ipdl.inpit.go.jp/Tokujitu/tjktk.ipdl
[5] 株式会社リコー，非視覚操作インタフェースの開発，https://www.ricoh.com/ja/technology/techreport/31/pdf/A3113.pdf
[6] 森田，板倉，“自己相関法による音声の時間軸での伸縮方式とその評価”，電子情報通信学会技術研究報告 EA86-5，pp.9-16，1986.