TR-IT-0342

自動ラベリングツールの評価

藤井 慶　北川 敏　Nick Campbell

2000.2

概要

ATR 音声翻訳通信研究所 第 2 研究室が開発した多言語音声合成システムに CHATR がある．CHATR への話者のデータベース作成は現在人手で行われており，長時間を要することが問題である．本報告では，この作業を自動化するためのラベリングツールの精度について説明し，その精度について述べる．またツールの改良案として無音区間の除去について説明する．

もくじ

第 1 章

はじめに

ATR 音声翻訳通信研究所が開発した多言語音声合成システムに CHATR がある．このシステムの音声合成は，データベースから音素ごとに最適な波形を選択し接続することで実現される．

合成時に必要となるデータベースは現在技術と経験を持つ人の手によって作成されている．そのため精度の高いデータベースが得られるが，反面，作成に長い時間を要するという欠点がある．そのため多くの話者データベースを短時間で作成するのは困難である．

現在 ATR には，将来話すことができなくなる患者 約 4000 人の声を CHATR のデータベースに登録して欲しいという要望が来ている．これは今のうちにデータベースを作っておけば，患者が将来話せなくなったときに，CHATR を用いて自分の声で話すことが可能になるためである．

しかし人手に頼っている現状では，それは時間の面から不可能である．そのためデータベース作成の自動化が必要になっている．

そこで，今回データベース作成の自動化を本研究の目標とした．これを実現する一手法として OSK 作成の自動ラベリングツールがある．このツールでは話者を登録するために，その登録話者と発話内容が同じデータを事前に用意し，それに正しいラベル付けをしておく．そしてそれを用いて登録話者の音声にラベル付けする．

本報告では，このツールの概要について説明し，その性能を調べた結果を報告する．また改良案として，ツールを用いる前に音声中の無音区間を削除する実験を行った．この結果についても述べる．

なお，最終的に想定するのは

1. 発話は 1 つのテープにまとめて収まっている．
2. 発話文は既知．また発話文の順序も既知．
3. しかし録音テープのどこで発話されているかは未知 (発話文の順序は代らない)．

という環境で自動でデータベースを作成することである．

第 2 章

OSK 自動ラベリングツール

2.1 処理の流れ

OSK 自動ラベリングツールの処理の流れを図 2.1 に示す.

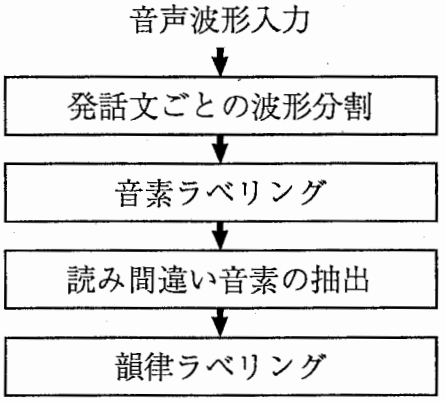


図 2.1: OSK 自動ラベリングツール

本研究で調査したのは音素ラベリング部までである. 各部について以下の節で説明する.

2.2 発話文ごとの波形分割

波形分割の処理の流れを図 2.2 に示す.

まず epda コマンドを用いてパワーを求める. そしてパワーがしきい値未満である区間長がしきい値以上である所を探し, そこに文の区切りがあると判断する.

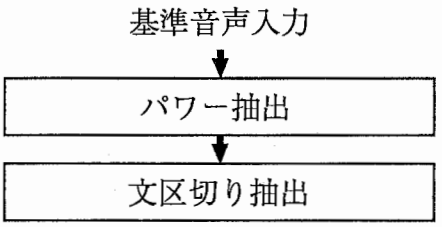


図 2.2: 波形分割

2.3 音素ラベリング

音素ラベリングの処理の流れを図 2.3 に示す.

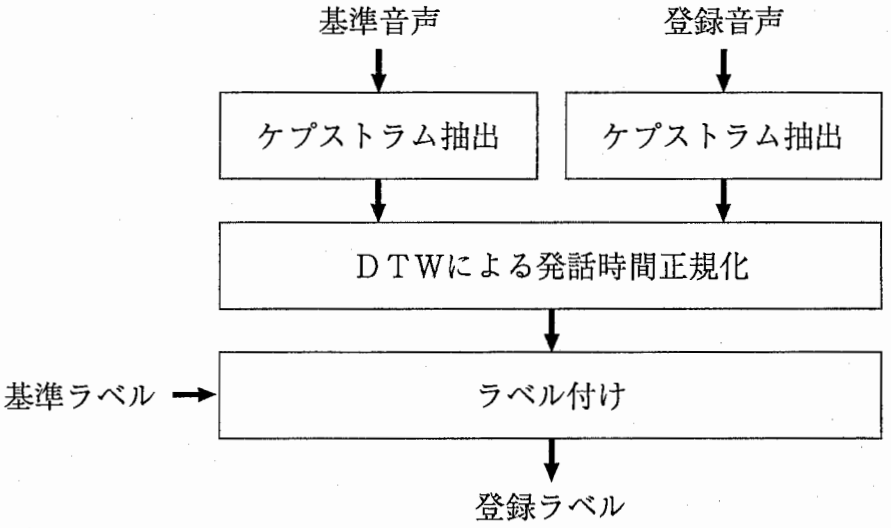


図 2.3: 音素ラベリング

まず音声からケプストラムを抽出する[1]. 以前は, ケプストラムは cepanaf を用いて抽出されていた. しかし cepanaf はサンプリング周波数 12000$[Hz]$ 固定であったため, 周波数可変の bispec を用いるよう今回変更した.

次に互いのケプストラムを特徴量として DTW を行い, 時間を正規化する. DTW でとりうる路を図 2.4 に示す.

最後に基準音声に正しく付けられた基準ラベルを, DTW の結果に基づいて登録音声に写す. この概念図を図 2.5 に示す.

[1]実際には, 発話文分割の前にケプストラム抽出を行い, 発話文分割の結果に沿ってケプストラムを分割している. しかし分割後に個別にケプストラム抽出を行う方がメモリ節約の面等で良いと考える.

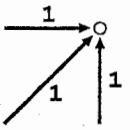


図 2.4: DTW のとりうる路

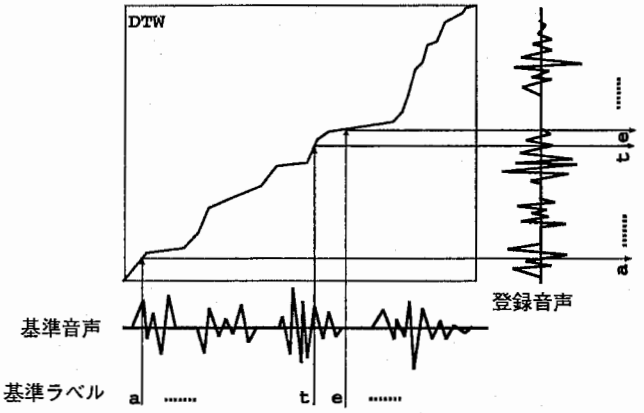


図 2.5: ラベル付け

2.4 読み間違い音素の抽出

読み間違い音素抽出は次の手順で行われる.

1. まずデータから同じ音素のケプストラムを抽出する (これらの音素は正しく発話されているとは限らないし, 正しい位置にラベル付けされているとも限らない).
2. 時間ごとに平均と標準偏差を求める.
3. ケプストラムの許容範囲を求める (範囲: (平均 − 標準偏差)$\cdots$(平均 + 標準偏差))
4. 音素中のケプストラムのうち, 許容範囲外にあるフレーム数を求める.
5. 許容範囲外にあるフレーム数がしきい値 pv より大きければ, その音素は正しく発音されていないか正しい位置にラベル付けされていないと考え, 廃棄する.

この流れを図 2.6 に示す.

しかし, この抽出法には次の問題点が挙げられる.

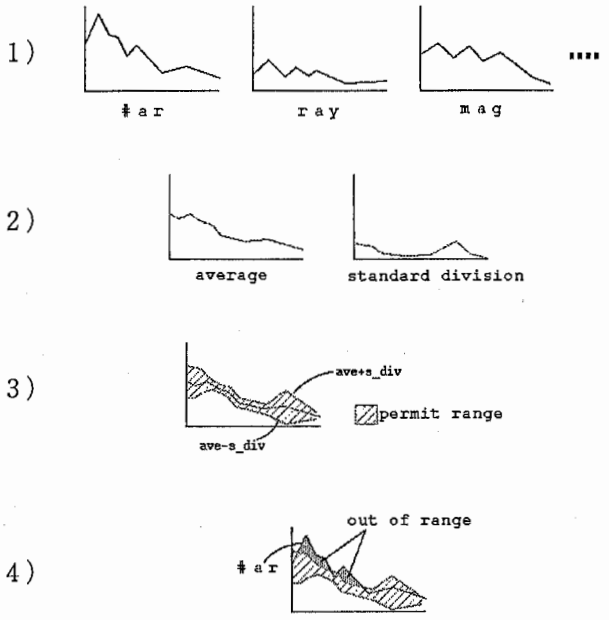


図 2.6: 読み間違い音素抽出

- **短い音素は間違いにされない**
  短い音素 (長さが pv 以下) は, どのようなものであろうと 5. で必ず正しい音素と判断されてしまう.

- **— 文中に 1 つしかない音素は間違いにされない**
  文中に音素が 1 つしかない場合, その音素のケプストラムが平均ケプストラムとなる. するとどのような音素でも上記 4. での許容範囲外の要素数が 0 になるため必ず正しい音素と判断されてしまう.

- **くせのある音素が廃棄される可能性がある**
  平均的な音素の形と違うものほど廃棄されやすい. そのため正しい音素でも平均的なものから離れている場合は廃棄され得る.

第 3 章

自動ラベリングツールの評価

3.1 評価方法

本節では, 自動ラベリングツールが付けた音素ラベルの精度の評価方法を説明する. 評価は, OSK が登録音声に付けたラベルと, 人が付けた理想ラベルを比較して行う. 両者の違いが小さいほど OSK の精度が高いといえる.

基準ラベルを $L_{S_1}, L_{S_2}, \cdots, L_{S_N}$, 登録ラベルを $L_{R_1}, L_{R_2}, \cdots, L_{R_N}$ とする. 本研究では, 次の条件を 1 つでも満たす登録ラベル L_{R_i} は正しく付けられていないラベルであるとみなす.

1. $Lag(L_{S_i}, L_{R_i}) >$ しきい値T
2. $Lag(L_{S_i}, L_{R_i}) > Lag(L_{S_{i-1}}, L_{R_i})$
3. $Lag(L_{S_i}, L_{R_i}) > Lag(L_{S_{i+1}}, L_{R_i})$

ここで, $Lag()$ は時間差を求める関数を示す. 1. はラベルの絶対的なずれに関する条件であり, 2.,3. は相対的なずれに関する条件である. 間違いラベルを求めるプログラムとして propererror.pl を今回作成した.

全ての登録ラベルに対してこの評価を行い, 間違いラベルの数を求める. 全ラベル数に対する間違いラベル数の割合が小さいほど OSK の精度が高いといえる.

3.2 評価実験

前節で述べた評価方法で自動ラベリングツールの付けたラベルを評価する実験を行った. 実験に用いた環境を表 3.1 に示す.

今回の実験では, 生の音声ではなく CHATR による合成音声を用いた. これは合成音声には雑音の混入が少なく, 発声も互いに大きく違うことがないためである. すなわち今回の実験は, ある

表 3.1: 評価実験

話者	MAK, MBY, MHM, MHN, MNC, MTH, MYA, MYS による合成音声
発話	「あらゆる現実をすべて自分の方へねじまげたのだ」
しきい値 T	0.05, 0.1

理想的な環境下での精度を擬似的に求めるために行った.

実験は話者を 2 人選び, 一方のデータを基準用, 他方のデータを登録用としてラベリングツールにラベル付けさせた. その結果を表 3.2, 3.3 に示す. 表内の数値は該当する組み合わせで得られた間違いラベル数である.

表 3.2: 実験結果 ($T = 0.05$)

	MAK	MBY	MHM	MHN	MNC	MTH	MYA	MYS
MAK	-	19	5	7	6	18	14	17
MBY	20	-	18	21	43	12	20	13
MHM	8	21	-	18	7	4	4	6
MHN	10	23	23	-	12	16	19	7
MNC	9	45	8	12	-	29	5	19
MTH	16	14	7	15	28	-	5	5
MYA	12	18	1	15	5	5	-	12
MYS	14	15	8	4	17	4	11	-

間違いラベル数 = 769, 間違いの割合 = 28.74%

結果として $T = 0.05$ のとき間違いの割合は 28.74%, $T = 0.1$ のとき間違いの割合は 28.25% となり, 3 割近くが正しい位置にラベル付けされていないことが分かった.

表 3.3: 実験結果 ($T = 0.1$)

	MAK	MBY	MHM	MHN	MNC	MTH	MYA	MYS
MAK	-	19	5	7	6	18	14	16
MBY	20	-	17	21	43	11	20	13
MHM	7	20	-	18	6	4	4	6
MHN	10	23	20	-	12	16	19	7
MNC	9	45	8	12	-	29	5	19
MTH	16	14	7	15	28	-	5	4
MYA	11	18	1	15	5	4	-	12
MYS	14	15	7	4	17	4	11	-

間違いラベル数 = 756, 間違いの割合 = 28.25%

第 4 章

自動ラベリングツールの改良案

4.1 無音区間が DTW に及ぼす影響

本章では自動ラベリングツールの精度を上げる一方法について提案する. 今回は, データに含まれる無音区間が DTW の結果に影響を与えることに着目した. 図 4.1 は基準音声の発話後に無音区間があるために DTW で求めたパスが理想の最適パスと大きくずれている一例である.

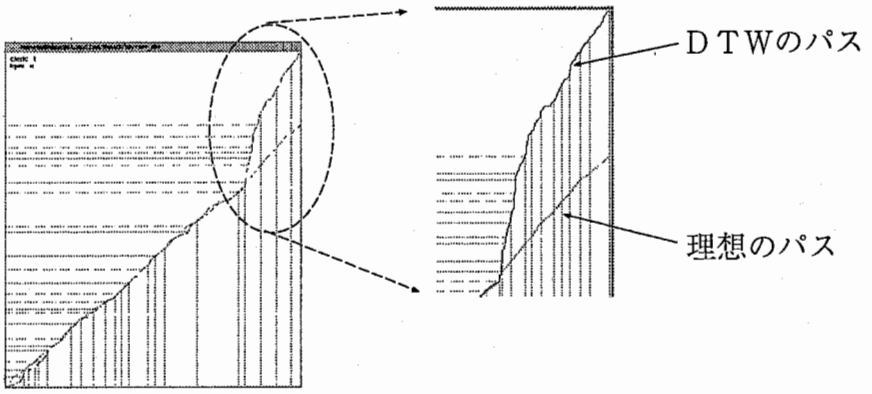


図 4.1: 無音区間があるときの DTW 結果例

そこで本章では, 事前に無音区間を削除した音声をラベル付けしたときの精度の変化について述べる.

4.2 無音区間の削除

本研究では, 音が無音区間であるかどうかをその場所のパワーがしきい値 P 未満かどうかで判断する. パワーは get_f0 コマンドを用いて求められる. 無音区間と判断された区間は音声から削除され, 同時にラベルもその時間だけ前に詰められる. 無音区間を求めるプログラムとして unvoice

を, 指定した区間を音声とラベルから削除するプログラムとして cut_unvoice.pl を今回作成した.

4.3 評価実験

無音区間を除去したデータを自動ラベリングツールにラベル付けさせた結果を評価した. 実験環境は 3.2 節での評価実験と同様である (表 3.1). 結果を表 4.1, 4.2に示す.

表 4.1: 実験結果 ($T = 0.05$)

	MAK	MBY	MHM	MHN	MNC	MTH	MYA	MYS
MAK	-	18	3	6	6	16	14	7
MBY	18	-	17	13	20	11	11	13
MHM	5	21	-	13	11	4	6	7
MHN	9	20	17	-	12	13	14	9
MNC	8	20	10	10	-	7	5	18
MTH	15	12	5	10	8	-	5	5
MYA	11	11	4	7	4	5	-	12
MYS	5	16	8	4	17	4	12	-

間違いラベル数 = 592, 間違いの割合 = 22.12%

表 4.2: 実験結果 ($T = 0.1$)

	MAK	MBY	MHM	MHN	MNC	MTH	MYA	MYS
MAK	-	18	3	6	6	16	14	7
MBY	18	-	16	13	20	10	11	13
MHM	5	20	-	13	10	4	6	7
MHN	9	20	14	-	12	13	14	9
MNC	8	20	10	10	-	6	5	18
MTH	15	12	5	10	8	-	5	5
MYA	11	11	4	7	4	5	-	12
MYS	5	16	8	4	16	4	12	-

間違いラベル数 = 583, 間違いの割合 = 21.77%

結果として $T = 0.05$ のとき間違いの割合は 22.12% となり, 従来の割合に比べ 6.62% だけ間違いが減少した. また $T = 0.1$ のとき間違いの割合は 21.77% となり, 従来の割合に比べ 6.48% だけ間違いが減少した.

4.4 問題点

今回の無音区間を削除する方法には問題点がある.

- 例：「ぶっか (物価)」→「ぶか」
  小さな「っ」は無音になるため, 削除される可能性がある
- 大きな雑音は除去されない
  音声と雑音の区別をしないため, しきい値 P 以上の雑音が除去されない.

第 5 章

まとめ

本報告では次のことを述べた.

- 自動ラベリングツールの性能評価として 2676 個の音素に付いて実験を行い, 正しく付けられなかったラベルの割合 $28.74\%(T = 0.05), 28.25\%(T = 0.1)$ を得た.
- ツールの改良案として事前に無音区間を除去する方法について述べ, 間違いラベルの割合 $22.12\%(T = 0.05), 21.77\%$ の結果, すなわち従来法に対して $6.62\%(T = 0.05), 6.48\%$ 分の改良結果を得た.

今後の課題としては次のことが挙げられる.

- 最適な基準パターンの選択

謝辞

今回非常に貴重な研究の機会を与えて下さり, 多方面に渡り適切な御指導を頂いた Nick Campbell 第二研究室室長に深く感謝致します. この 2 ヶ月間は非常に有意義な経験となりました.

また研究内容にとどまらず様々な助言を頂いた ATR 音声翻訳研究所第二研究室の皆様に深く感謝致します。

1999 年 9 月 24 日

藤井 慶

付録 A

作成したプログラム

A.1 各プログラムの概要

今回の研究で作成したプログラムの概要を表 A.1 に示す.

表 A.1: 作成したプログラム一覧

プログラム名	用途
cut_unvoice.pl	指定した無音区間を音声・ラベルから削除する (4.2 節を参照)
divide	音声ファイルをラベルに従って音素ごとに再生する
propererror.pl	ラベルが正しく付けられているか検査する (3.1 節を参照)
timelag.pl	2 つのラベル間の時間差の平均・最小値・最大値・ヒストグラムを出力する
unvoice	無音区間を求める (4.2 節を参照)
view_cep	bispec で求めたケプストラムをフレームごとに表示する
view_dtw	ツールが行った DTW の結果を表示する

プログラムは /home/as69/xkfujii/LabelTool/Source/Instant/ の下にある. 次節以降で各プログラムについて説明する.

A.2 cut_unvoice.pl ... 指定した無音区間を音声・ラベルから削除する

機能

指定した区間を音声ファイル・ラベルファイルから削除する.

インストール

1. 1 行目に perl を絶対パスで指定する.

2. chmod timelag.pl

実行

% cut_unvoice.pl rawfile(in) rawfile(out) [labelfile(in) labelfile(out)]

rawfile(in) 入力音声ファイル

rawfile(out) 出力音声ファイル

labelfile(in) 入力ラベルファイル

labelfile(in) 出力ラベルファイル

※ラベルファイルは指定しなくてもよい

実行例

% unvoice A.raw | cut_unvoice.pl A.raw A_cut.raw A.lab A_cut.lab

A.raw から無音区間を求めて削除し, 結果を A_cut.raw, A_cut.lab に出力.

操作

削除する区間は標準入力から指定する. 指定方法は次の命令を用いる.

- **from** A **to** B　　A から B までを削除
- **from** A　　A から末尾まで削除
- **to** B　　先頭から B までを削除

命令は 1 行につき 1 つ.

入力終了は [ctrl]-d.

留意点

音声ファイルの書き出しは 16 ビットずつ行っている. そのため実行速度が遅い.

A.3 divide ... 音声ファイルをラベルに従って音素ごとに再生

機能

指定した音声ファイル・ラベルファイルを音素ごとに再生する.

インストール

% gcc divide.c -o divide

実行

% divide [-s freq | -a] labelfile rawfile

-s freq サンプリング周波数

-a 分割した音を 0.4 秒刻みで再生していく

labelfile 入力ラベルファイル

rawfile 音声ファイル

操作

オプション -a を付けない場合, 再生は対話的に行う. その時の操作には次のものがある.

- **改行** 音素を再生し, 次の音素へ.
- **p, play** 音素を再生.
- **n, next** 次の音素へ.
- **q, quit** 終了.

留意点

音声ファイルの読み書きを 16 ビットずつで行っている. そのため実行速度が遅い.

A.4 propererror.pl ... ラベルが正しく付けられているか検査する

機能

3.1 節で述べた評価基準でラベルを検査し, 間違いラベルを 「X」 に置き換えて出力する.

インストール

1. 1 行目に perl を絶対パスで指定する.

2. chmod timelag.pl

実行

% propererror.pl labelfile(ideal) labelfile(for check) [labelfile(in) labelfile(out)]

labelfile(ideal) 正しく付けられたラベルファイル

labelfile(for check) 正しく付けられているか検査するラベルファイル

実行例

% propererror.pl A.lab A_osk.lab | grep X | wc | gawk '{print $1}'

A_osk.lab の間違いラベル数を求める

留意点

時間差のしきい値 T を実行時に変更するオプションがない.

A.5 timlag.pl ... 2 つのラベル間の時間差の平均・最大・最小・ヒストグラムを表示

機能

2 つのラベル間の時間差の平均・最大・最小・ヒストグラムを表示する.

インストール

1. 1 行目に perl を絶対パスで指定する.

2. chmod timelag.pl

実行

% timelag.pl [-i interval] labelfile1(ideal) labelfile2(for check)

-i interval ヒストグラムの間隔を指定する.

labelfile1 正しく付けられているラベルファイル

labelfile2 正しく付けられているか検査するラベルファイル

A.6 unvoice ... 無音区間を求める

機能

4.2 節で述べた方法で無音区間を求め, その時間を「from A to B」の形式で出力する

インストール

% make

実行

% unvoice [-p threshold | -f frequency | -v] rawfile

-p threshold 無音かどうかを判断するしきい値.

-f frequency サンプリング周波数

-v 途中経過の表示

留意点

- unvoice はパワーを求めるために btosps, get_f0 コマンドを用いる. そのため実行時にはこれらのコマンドにパスが通っている必要がある.
- /DB/PI/{OSNAME}/bin/unvoice とは別物である.

A.7 view_cep ... bispec で求めたケプストラムをフレームごとに表示

機能

bispec で求めたケプストラムをフレームごとに表示する.

インストール

- Makefile の INCDIRS と LIBDIRS に X11 のディレクトリを指定する.
- % make

実行

% view_cep [-fl framelength | -e num | -c num] cepfile

-fl framelength ケプストラムのフレーム長を設定. デフォルトは 0.001 秒

-e num 1 フレーム内の要素数を設定. デフォルトは 16 個

-c num フレームを同時に幾つまで表示できるかを設定. デフォルトは 4 個

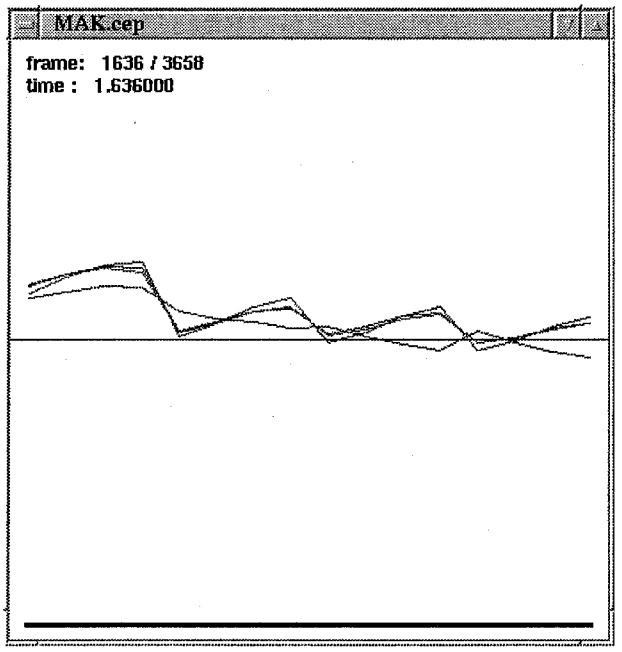
操作

操作は主にマウスで行う.

- **右ボタン** 次のフレームへ
- **左ボタン** 前のフレームへ
- **中ボタン or 'q' を押す** 終了
- **画面下方の黒太線にポインタを会わせる** ポインタが指した位置のフレームを表示

実行例

% view_cep A.cep



赤線 現在位置のケプストラム

緑線 前のケプストラム. -c 1 で非表示.

1636 現在位置

1.636 現在位置の時刻 (秒)

3658 フレームの総数

A.8 view_dtw ... DTW の結果と理想のパスを表示

機能

ツールを -g tmp 付きで実行したときに残される DTW の結果ファイル dtw0000.dat を基に DTW の求めたパスを表示する．また理想ラベルファイルを指定することで理想パスも表示する．

インストール

- Makefile の INCDIRS と LIBDIRS に X11 のディレクトリを指定する．
- % make

実行

% view_dtw [-fl framelength | -v] dtwfile [labelfile(ideal) [labelfile(check)]]

-fl framelength ケプストラムのフレーム長を設定．デフォルトは 0.001 秒

-v 途中経過を表示

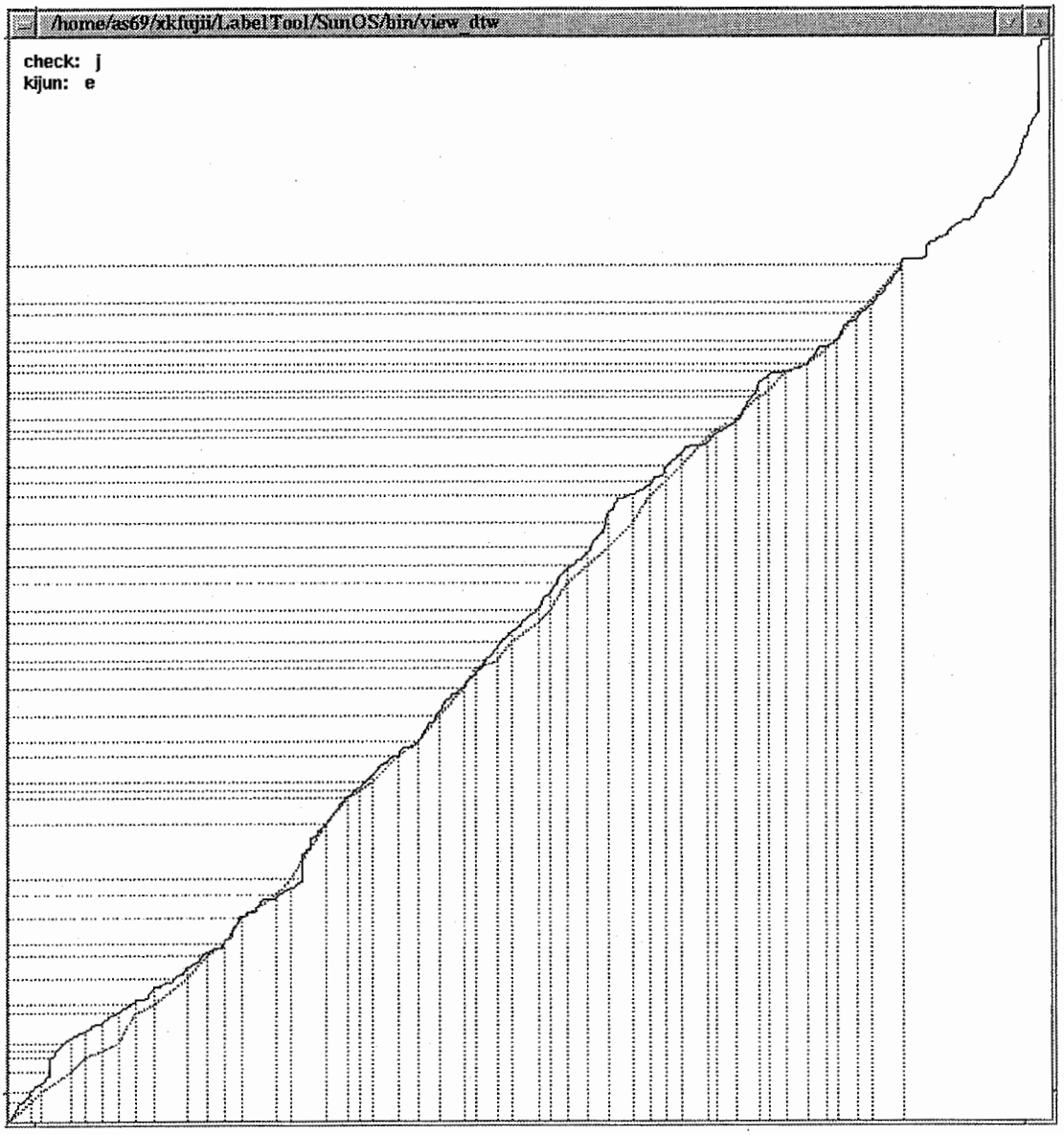
操作

画面内でマウスボタンあるいはキーを押すと終了する．

マウスポインタのある位置の音素が画面左上に表示される．

実行例

% view_dtw dtw0000.dat A.lab A_osk.lab



黒実線 DTW の求めたパス

緑点線 理想のパス

青点線 基準ラベル

赤点線 チェックラベル

黒実線と緑点線が近いほど良い結果といえる.

留意点

- ラベリングツールは DTW を行う前に発話文を分割するが, view_dtw は文が分割されなかった (文が 1 つだった) ときのみ正しく作動する.