TR-IT-0210

音声合成システム"CHATR"における音声単位選択アルゴリズムの改良

梅田一郎,Nick Campbell

1997 3.

ABSTRACT

音声合成システム CHATR の音声単位選択アルゴリズムの改良を検討した。結果、

- continuity distance における pitch の割合を上げることで f0 の跳躍による聞きづらさを軽減できた.
- VQ により最適接続部を求めるルーチンに訂正を行い，接続性を上昇させた.
- continuity distance を melcepstrum の VQ と cepstrum により求め，音声単位候補選択時の context 及び duration への依存度を下げた．よって，より的確な音素を候補に上げることが可能になった.

Contents

Chapter 1

始めに

現在エイ・ティ・アール音声翻訳通信研究所では，音声合成研究の一環として多言語音声合成システム CHATR の研究を行なっている．

このシステムは，大量の音声データベースの中から最も近い音声波形を抽出し接続するものであり，信号処理を行わないために非常に自然な音声が得られる反面，非常に大量のデータベースを要求すること，データベースが完全でない現時点では出力音声の f0 や power が著しく入力と異なる事，等の弊害もある．

よって，音声波形の単位選択アルゴリズムは，CHATR における合成音声の自然性を左右する重要なファクターである．既に，最も近い音声波形を抽出する為のいくつかのアルゴリズムがあるが，十分ではない．

本研究員はこの研究の一端を担い，音響特徴や韻律特徴を調べながら Viterbi アルゴリズムの高度化をサブゴールとした．また，CHATR の持つ多くのパラメータを操作して単位選択に対する影響を計測・検討し，この実験結果より韻律特徴や音響的特徴の役割によって，単位選択アルゴリズムを書き直し，高度化した．

Chapter 2

音声単位選択部の概要

本章ではCHATRの音声単位選択部アルゴリズムの概要及び問題点を述べる。

2.1 音声単位選択方式とは

音声単位選択という考え方はCHATRの前身のATR-ν Talkに遡る.

まず，ある話者の朗読音声を集めた音声データベースを用意し，このデータベースは，対象とする言語に表れる全ての音素をカバーする物とする.

合成時には，合成するべき音素列を入力とし，この音素列からf0やpowerを生成する.

入力ストリームの各音素に対し，データベース内から同じ音素を検索し，このデータベース内の各音素の音響・韻律特徴と，入力ストリームで得られる音響・韻律特徴の差を求める．これをunit distanceあるいはunit costとする． unit costが小さいものほど入力ストリームが求める音素にふさわしい音素と言える.

合成はこれらのデータベース内の音素を接続していく事で行うが，接続時にはf0やpowerの跳躍，音声スペクトルの相違などによる聞きづらさが生じる．そのため，音素間のf0 , power , cepstrum / melcepstrum等の差を計上し，これを候補間のcontinuity distance (もしくはjoin cost)とする． continuity distanceが小さいものほど，前候補との接続がスムーズに行われる音素と言える.

この二つのdistanceを考慮し重み付けを行い，最も両distanceの総和の小さい音素を選び，接続する．すなわち， ATR-ν talkの音声単位選択アルゴリズムは音素を「街」，接続を「道」とした最適経路問題に帰着する． (Figure 2.1)

そのため， CHATR内でも，音響的側面から見たdistanceと，最適経路問題としての側面からみたcostと，同じ意味を指す二つの用語が混在している．本稿は基本的にソースプログラムにおける関数名・変数名に準拠している.

以上がATR-ν talk及びCHATRの基本的な合成法である.

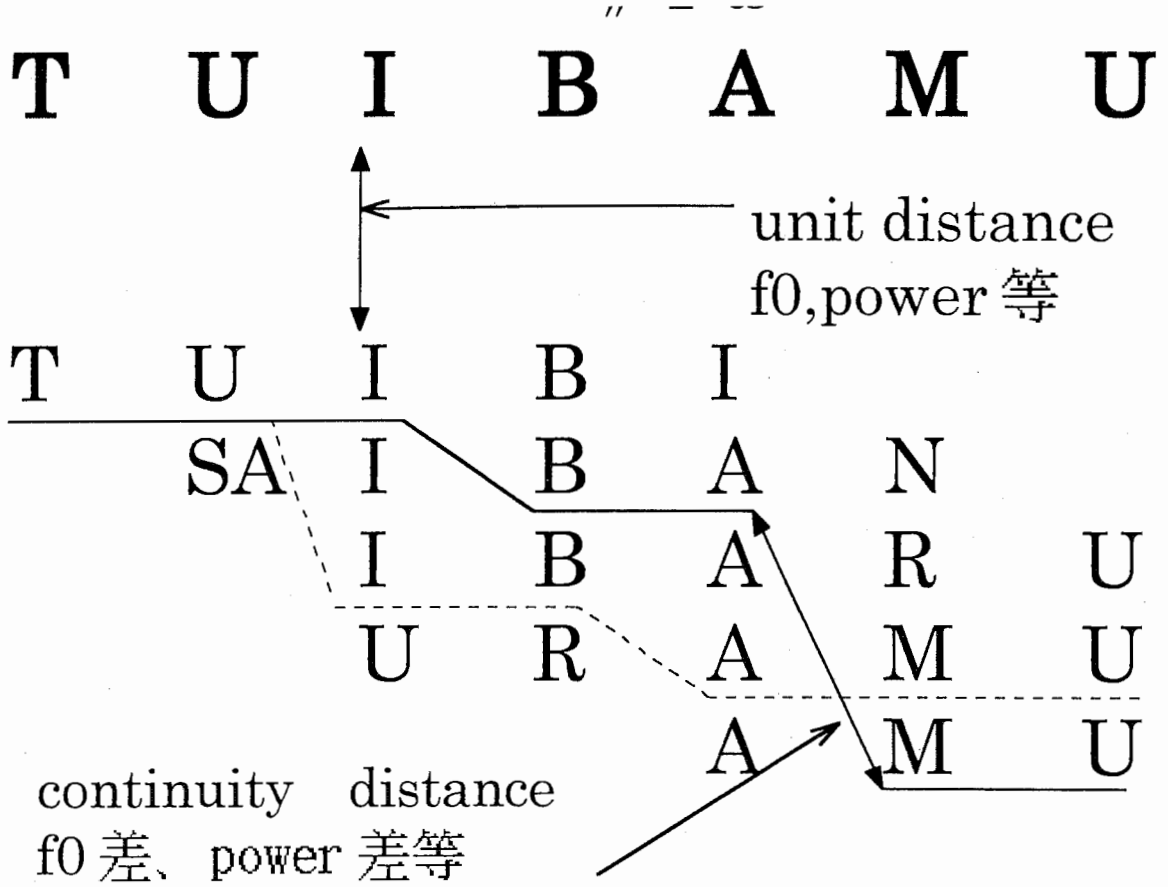


Figure 2.1: 音声単位選択の概念図（ATR-ν Talk）

2.2 find_best_path

入力音素列やデータベースからの音素候補群を基に最適経路問題を解く部分である.

2.2.1 プログラムのソースの抄訳

```
static void nus_find_best_path(入力 stream)
{
     for (c = 入力 stream 全音素について)
    {
        /* First find some candidates for target */
        nus_find_unit_candidates(c);
            /* Path ごとに候補は生成すべきでは？ */
        nus_prune_paths(c);  /* Prune paths to this point */
        nus_prune_cands(c);  /* Prune candidates at this point */
        /* For each path to score joins with candidates */
        for (p = 全ての path 候補について)
        {
            for (cand = 全ての音素候補について){
            {   /* find join cost for each possible extension */
                cont_dist = nus_continuity_distance(前音素, 候補音素);
                newcost = この path のこれまでのコスト +
                    (cand->unit_distance * nus_unit_weight) +
                    (cand->continuity_distance * nus_join_weight);
                nus_make_new_path(c,p,cand,newcost,cont_dist);
            }
```

```
        }
    }
    return;
}
```

2.2.2 プログラムの解説

まず，入力音素列内のある音素に着目し，同じ音の音素をデータベースから検索して， unit distance の低い順に音素候補群を生成する．その後，全 path 候補に対し，音素候補それぞれとの continuity distance を求め， path を更新する．その後、 path に対し枝狩り (pruning) を行う．

以上を全ての入力音素に対し行い，最終的に distance の総和が最小であるものを最適解とする．(Figure:2.2)

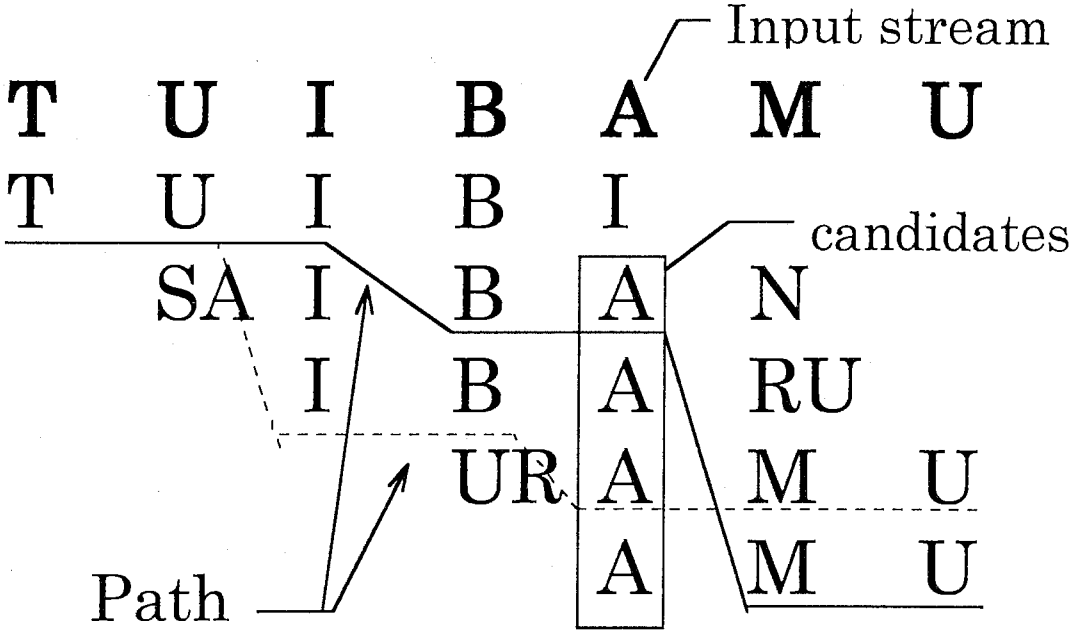


Figure 2.2: find_best_path 概念図

2.2.3 問題点

ここにはいくつかの問題点が存在する.

- BEAM search による one-pass DP では，枝狩りを行なった時に，最適解を導かない可能性がある.
- 枝狩りを伴う DP アルゴリズムでありながら左 context に強く依存するために robust とは言い難い.
- path A の最適解である音素と path B の最適解である音素は異なり，候補群を生成した時点でいずれかの path の最適解が含まれていない可能性がある.

これらの検証結果は次章で行う.

2.3 find_unit_candidate

入力 stream の文脈情報に基づき，音声データベースから音素の候補群を選別する部分である.

2.3.1 プログラムのソースの抄訳

```
static void nus_find_unit_candidates(注目音素)
{   /* Search context index for best n matches */
    /* Find their unit distortion from target  */

    /* Find the phoneme context for this target */
    ug_get_stream_context(注目音素,context テーブル);

    /* For each occurrence of ph in the database */
    for (idx = データベース内の全ての同音素について){
    {
        cost = ug_check_context(context テーブル, 注目音素);
        if (十分に近いなら){
            /* 最適解が棄却されないか？ */
            予備候補群に登録
            cand->unit_distance = cost;
            /* 根拠が薄いのでは？ */
        }
    }

    /* Do full analysis and make the candidate list */
```

```
    for (ic= 全ての予備候補群音素について){
    {
        if (十分にデータが存在する  && 前後の音素が遠い){
            break;

        予備候補群から本候補群にコピー
        ic->cost = udb_unit_distance_df(注目音素、 ic);

        cand->unit_distance += nus_lcontext_weight * ic->cost;
    }
    return;
}
```

2.3.2 解説

まず，データベースごとに固有の，音素間の音響的距離（類似性）を格納した配列を読み込む.

データベース内の着目した音素と同じ音の音素全てについて，入力 stream とデータベースの音素の前後 2 音素づつの音響的な距離を元に，注目音素からの時間的重みを基に重みを付加して加算し， context cost とする.

context cost の小さい方から最大候補数個分を候補群に加えるが，候補群の個数が最小候補個数を越えていて，かつ， context cost がある程度以上大きければ棄却する.

候補群の中のそれぞれの音素について， unit distance を求め，それに， context cost を加算する. (Figure:2.3)

2.3.3 問題点

ここにもいくつかの問題点が存在し，次章で検証結果を示す.

- context による重みは母音・子音の区別等を考慮に入れておらず，選別の基準としては不十分である.
- 同様の理由で unit distance の要素の一員として不適当である.

- 間違ったスコアリングによって，候補選別時にある path の最適解が棄却されている可能性がある.

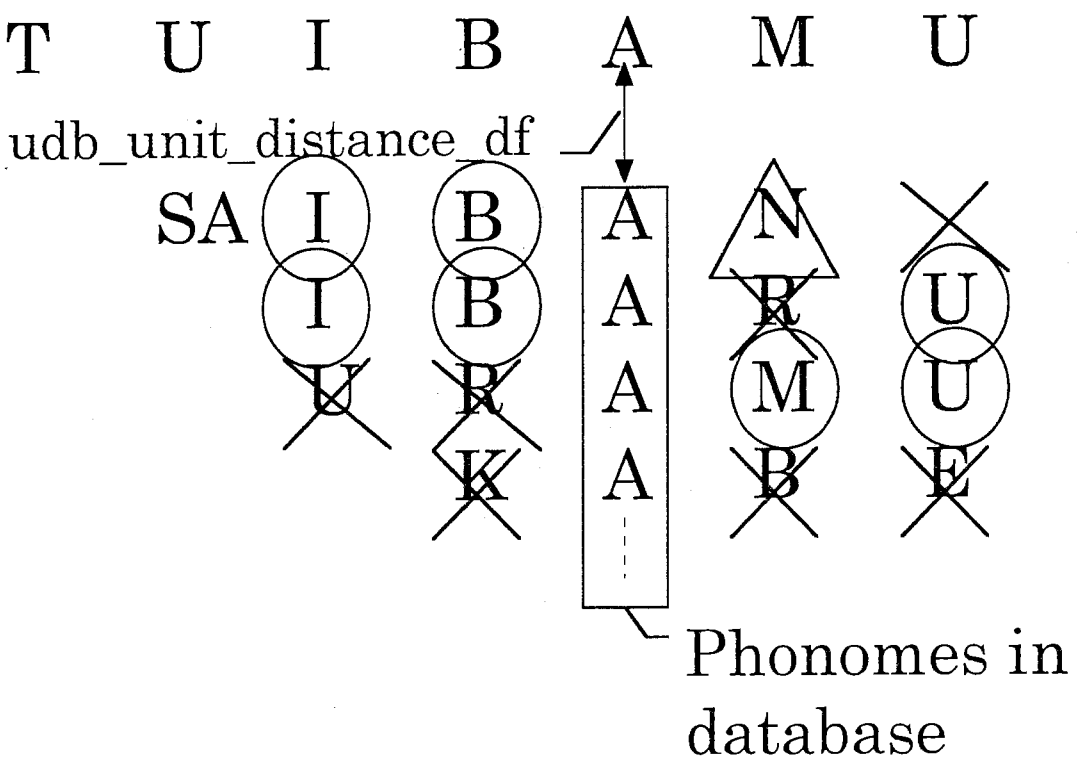


Figure 2.3: find_unit_candidate 概念図

2.4 udb_unit_distance_df

2.4.1 解説

入力 stream によって与えられた情報，あるいは入力音素列から音韻モデル等によって導かれた情報と，データベース内の候補音素との，

- duration
- pitch(f0)
- power
- f0-slant

などの距離を求め，各々重み付けして加算し， unit distance とする． (Figure:2.4)
距離を求めるに際し，単位・尺度などは下位 function にて考慮されている．

2.4.2 問題点

同じくいくつかの問題点が存在し，次章で検証結果を示す．

- 合成時に母音・ pause 等では音素データの一部分を切り出して用いることを認めることで、音素候補の柔軟性が広がる一方で、母音・ pause 等における duration の意味は希薄になると思われる．
- これらの各要素は互いに独立であるのか、単純加算による重み付けで構わないのか．
- 重み付け係数を定数ではなく関数にする事で，学習が容易になると思われる．

2.5 nus_candidate_distance

接続先音素（の末尾）と，それに接続したい音素（の先頭）との，音響的距離を求める部分である．

2.5.1 解説

接続先音素と当該音素との，

- phone
- prosody
- cepstrum
- vq

などの距離を求め，各々重み付けして加算し， continuity distance とする．現状では vq のみが使用されており， vq は 12 次元 melcepstrum,f0 及び power よりなる． (Figure:2.4)

距離を求めるに際し，単位・尺度などは下位 function にて考慮されている．

2.5.2 問題点

同じくいくつかの問題点が存在し，次章で検証結果を示す．

- 使用する vq アルゴリズムに改善すべき点が見受けられる．

2.6 コストマップ

以上見てきたように，階層構造になっているために，コストの加算表を作ることが出来る．(Figure:2.4)

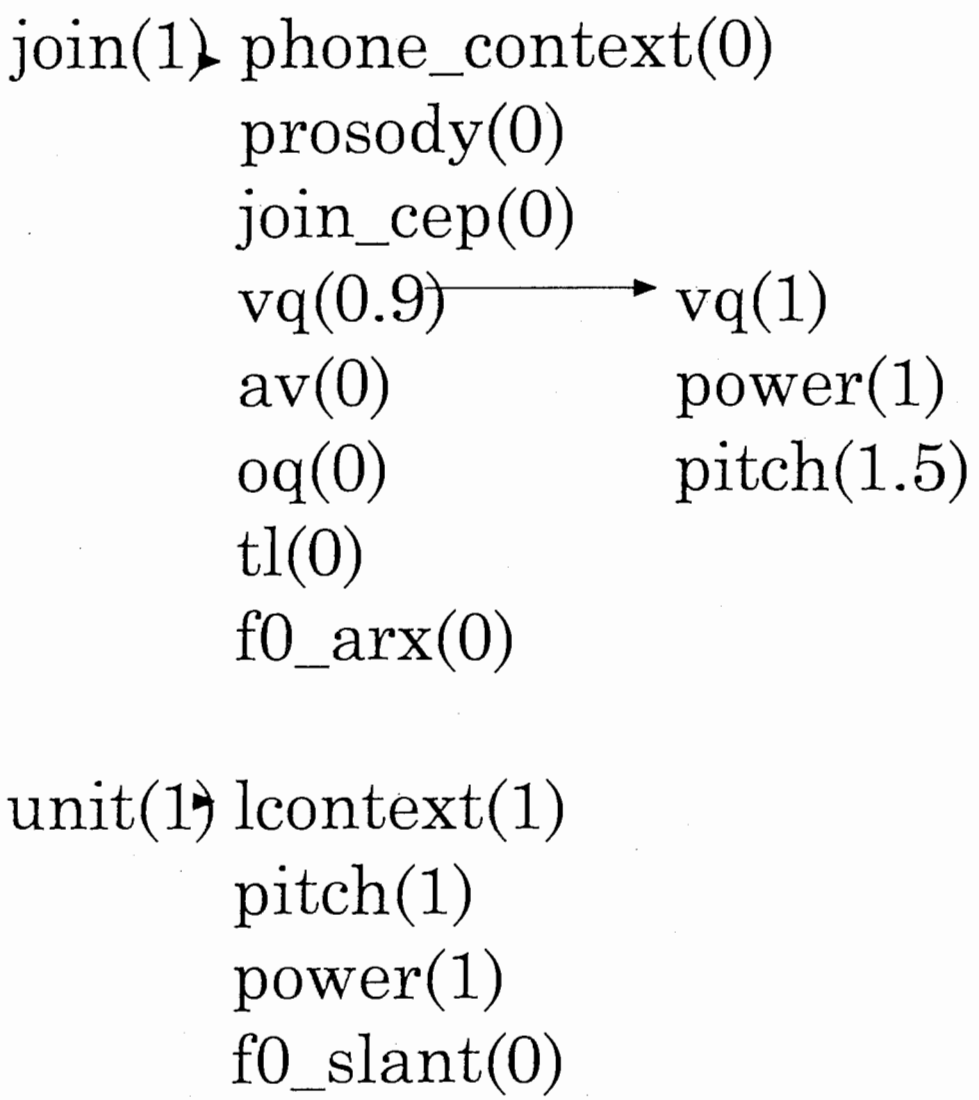


Figure 2.4: コストマップ

横方向に構成要素を示し，縦方向に加算を行なう．

2.6.1 問題点

いくつかの問題点が存在し，次章で検証結果を示す．

- unit distance に比べて join distance が小さい
- join distance 内の pitch の重みが小さい

- unit distance 内の lcontext は本来 join distance に属するべき物である.
- unit distance 内の pitch の重みが小さい

Chapter 3

CHATR の改良

前章で述べた問題点をまとめると以下のようになる.

- DP アルゴリズムの最適解の検証
- path ごとに音素候補を生成してみる
- context と duration による音素候補の選別の妥当性の検証
- continuity distance の一部としての context の妥当性
- VQ に関する訂正
- 各パラメータの最適値
- 各 distance の重み係数に関する部分

このうちのいくつかは実習期間中の研究が不可能であったため，まとめの章で予測と共に述べることとする.

3.1 各種パラメータの変更

音声単位選択アルゴリズムに関わるパラメータの変更の多くは外部から設定ファイルを変更する事で可能である。そこで、設定ファイルを変更して単位選択に対する影響を計測した。

3.1.1 nus_params.ch

現状の音声単位選択アルゴリズム周辺のパラメータは以下のようになっている。

```
(set nus_params
        '((beam_width 25)
         (cand_width 100)
         (join_wt 1.0)
         (unit_wt 1.0)
         (dur_wt default)
         (pitch_wt default)
         (power_wt default)
         (f0_slant_wt default)
         (lcontext_wt 1.0)
         (pp_phone_wt 1.0)
         (p_phone_wt 8.0)
         (n_phone_wt 4.0)
         (nn_phone_wt 1.0)
         (cand_thresh 1.0)
         (vq_wt 0.9)
         (vq_f0_wt 1.5)
         (vq_pow_wt 1.0)
         (phone_context_wt 0.0)
         (pros_context_wt 0.0)
         (dur_penalty 1.0)
         (endpoint_weight 0.0)
         (join_av_wt 0.0)
         (join_oq_wt 0.0)
         (join_tl_wt 0.0)
         (join_f0_arx_wt 0.0)
))
```

3.1.2 問題点

以下に問題点を述べる。

- 候補音素生成時のスコアリングの変更がある以上、beam_width , cand_width の数を増やしておくべきである。
- join weight が小さい。

- unit distance , continuity distance 共に power に比して pitch が小さい。特に unit distance では context による影響を減らす必要がある。
- continuity distance における vq の値が小さい。

3.1.3 変更後の nus_params.ch

変更後の nus_params.ch は以下の様になった。あくまで暫定的なものであり、また、客観的な評価は行なっていない。

```
(set nus_params
        '((beam_width 500)
          (cand_width 1000)
          (join_wt 2.0)
          (unit_wt 1.0)
          (dur_wt default)
          (pitch_wt 10.0)
          (power_wt default)
          (f0_slant_wt default)
          (lcontext_wt 1.0)
          (pp_phone_wt 1.0)
          (p_phone_wt 8.0)
          (n_phone_wt 4.0)
          (nn_phone_wt 1.0)
          (cand_thresh 2.0)
          (vq_wt 0.9)
          (vq_f0_wt 1.5)
          (vq_pow_wt 1.0)
          (vq_vq_wt 2)
          (phone_context_wt 0.0)
          (pros_context_wt 0.0)
          (dur_penalty 1.0)
          (endpoint_weight 0.0)
          (join_av_wt 0.0)
          (join_oq_wt 0.0)
          (join_tl_wt 0.0)
          (join_f0_arx_wt 0.0)
))
```

3.2 DP アルゴリズムの検証

nus_params.ch の beam_width と cand_width を増減させることで、DP アルゴリズムの精度を検証した。結果、いくつかの将来的な問題点は残るものの、現時点では最適解が求められていた。

3.2.1 問題点

CHATR は最適経路問題の解法手段として，BEAM search を用いている．ここで時間的・空間的な制限の為に随時枝狩りを行なっており，もし十分な作業領域が得られていないのならば，不十分な枝狩りが行なわれていることになり，最適解に至る枝を狩り，最適解を導かない可能性がある．

具体的には，例えば、現時点の話者 FMP の持つ a の音素は 3000 以上存在し，一方，最大候補数が 50，最大枝数が 25 である．すなわち，3000 個の音素から 50 個の候補を選ぶ過程，及び，最大候補数分存在しうる枝数から 25 本の枝まで刈る過程の 2 個所で，不完全な枝狩りを行なっている可能性がある．

一方、最大候補数及び最大枝数が十分に大きければ、BEAM search であるから、スコアリング法が変わらない限り、常に同じ最適解を導出するはずである。

3.2.2 実験

以上の事項の検証のために、最大候補数と最大枝数を変化させて，現時点でのデータベースにおける DP アルゴリズムの精度を調べた．この精度はデータベースの大きさに依存し，あくまで現時点での結果に過ぎないが，一方，データベースの大きさが増えた場合でも，最大候補数及び最大枝数の増加のみで DP アルゴリズムの精度は保証される．

3.2.3 結果

候補数	1000	100	50	38	25
枝数	500	50	25	19	13
FMP 正答率	100%	100%	100%	98.58%	90.28%
MKI 正答率	100%	100%	100%	94.04%	82.60%

話者 FMP 及び話者 MKI のデータベースにおいて，候補数を 1000 , 100 , 50 , 38 , 25 , 枝数を 500 , 50 , 25 , 19 , 13 と変化させ, CHATR demonstration の HK_00.ch ～ HK_11.ch を合成させた時に，最大候補数が 1000 の時と同じ音素 unit が選択された確率である．

ここで，FMP は朗読音声のデータベース，MKI は自然発話音声のデータベースである．

3.2.4 考察

この実験によって，現在のアルゴリズムである限りは十分な精度を得られていると言える．また，path ごとの最適解が欠落しているという事もない．

非最適解は連続して生じる傾向があり，単純に音素 unit 単位の誤答率で比較する事の根拠は薄弱ではあるが，一方，

- 候補数もしくは枝数が不足した時の非最適解率の急激な上昇
- 自然発話音声では朗読音声に比してさらに非最適解率の上昇が大きい

等の傾向が見受けられ，今後データベースの巨大化・汎用化に向けて注意が必要であると思われる．

また， burst 的に非最適解が生じると言う事は，不完全な枝狩りが行なわれた場合に，音素 unit 数個分に渡って影響が残るという事である．これは，左 context 重視による BEAM search 方式の根本的な欠点を提示しており，この方式による最大候補数・最大枝数の不足の深刻さを示している．

3.3 unit distance における context の妥当性

context 情報による音素候補選別部に音素数による重みを破棄し、有声無声の概念を導入した。

3.3.1 問題点

CHATR は最大 3000 を越える音素データベースの中から候補音素を選別し，それらの音素を接続して音声を合成する．この音素候補を選別する部分で，注目音素の前後 2 音素の context 情報を使用し，それぞれの音素の入力ストリームとの音響的距離を求め，注目する音素からの音素数によって重み付けしている．

ところが現在のアルゴリズムは，

- 母音子音や有声無声の区別が行なわれていない，
- 二つ前の音素が一つ前の音素に与える影響や，一つ後の音素が二つ後の音素に与える影響などが考慮されておらず，注目する音素からの音素数による重み付けでは不十分だと思われる．
- 音素の duration が考慮されておらず，例えば前音素が非常に長い duration を持っていたとしても，常に前前音素の重みは同じである．

などの欠陥を含む．

3.3.2 対策

注目する音素からの音素数による重みづけを廃止し、母音子音や有声無声の区別は，データベース内パラメータの無声 (0) ～有声 (1) により近似した．ただし、この数値に尺度的な保証はない。

また，隣接する音素同士の影響は，有声度の積を求め、その値を重み付けとする事で対処した．3.1

一方，duration 情報は現状では対応できていない．

入力	U	I	B	A	M	
database	A	I	B	A	N	
位置重み	$\frac{1}{14}$	$\frac{8}{14}$		$\frac{4}{14}$	$\frac{1}{14}$	
一致性	0.3	1	1	1	0.7	
改良前 score	0.214	0.571		0.285	0.05	1.120
有声度	0.8	0.9	0.6	0.9	0.2	
有声度累積重み	0.432	0.54		0.9	0.18	
一致性	0.3	1	1	1	0.7	
改良後 score	0.130	0.540		0.900	0.126	1.696

Figure 3.1: 有声度と累積による重み付けの概念図

3.4 unit distance における duration の妥当性

母音とポーズの場合に、音素の一部分の切り出しを認めてみたところ、特に pause における著しい改善がみられた.

3.4.1 問題点

入力音素列から韻律モデル等によって導かれた duration, もしくは直接入力で得られた duration と, データベース内音素の duration との差が unit distance に計上されている.

しかし, 母音やポーズ, 無声子音等では, 音素内の任意の部分を切り出して使用することが理論上は可能である. そのため, 以上の音素の場合には、候補の duration の方が入力の duration よりも長い分には、この音素は候補となりうる. (Figure:3.2)

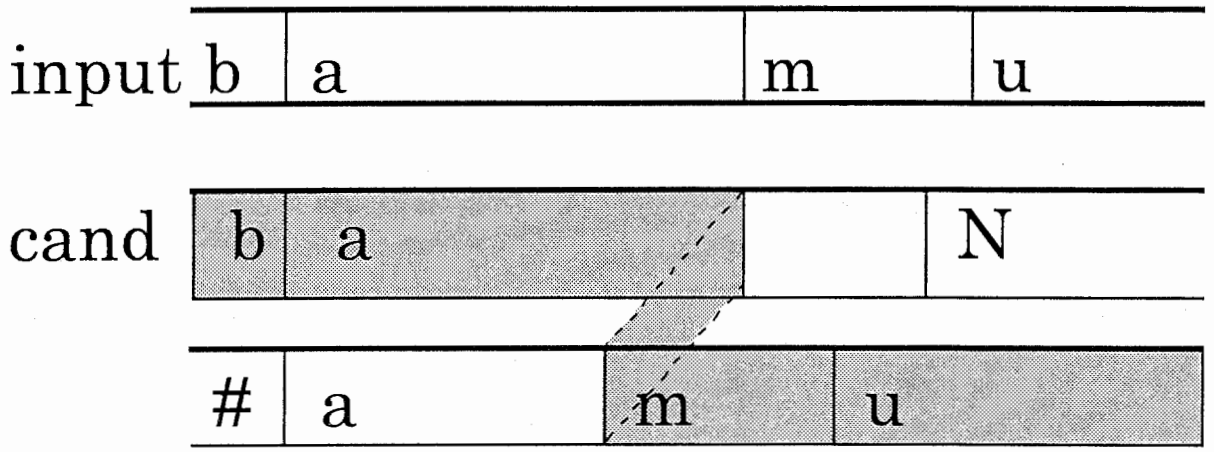


Figure 3.2: duration の切り出しの概念図

3.4.2 対策

ポーズもしくは母音の場合に候補の duration の方が入力の duration よりも長い場合には, unit distance 計上時の duration の重みを軽くした.

それとともに, continuity distance を求めるルーチン内部の, VQ による最適接続部検索時に, duration による切りだし部を付加した.

3.4.3 結果

これにより, 特に pause における著しい改善がみられた.

一方、無声子音の場合にはほぼ必ず母音が後続し, 切り出しの効果はほとんど認められなかった.

3.4.4 考察

この方法には多くの問題点が残る.

- 切り出し部は概念上 continuity distance ではなく unit distance 部に存在すべきである
- 右の音素候補のパラメータを必要とするために，現状の左 context 依存型の BEAM search では最適解が得られない
- 音素の切り出しを最大限に認めるならば，一つの入力音素を一つのデータベース内音素で担う必要はなく，複数の音素の接続で合成する事も可能であり，現在の、音素を最小単位とする合成系では，これは不可能である

3.5 continuity distance における context の妥当性

3.5.1 問題点

unit distance には前後 2 音素の context の音響的距離による score が計上されている．が，本来これは continuity distance に分類されるべきものであり，また， continuity distance の音響的要因の一つとして捕らえるには，根拠に欠けている．

3.5.2 対策

よって， context distance の unit distance への計上をやめたところ，著しく聞きづらさが増大した． continuity distance の重みを増やしても改善にはならず，現状の continuity distance の計算法に問題がある事がわかる．

continuity distance は，現状では melcepstrum による VQ 量子間距離と power,f0 によって求められており，この VQ の量子数が少な過ぎて精度が不十分だったことが原因と思われる．

そこで cepstrum の距離も continuity distance に導入したが、まだ十分とは言い難く、continuity distance における VQ の重みを増やす事で対応した。

本来は VQ の量子数を増加させることが望ましい。

3.5.3 考察

今後， context の distance に音素種別（舌・唇・声道の状態，発音源，等）の概念を導入する事で，音響的特徴量のパラメータとして unit distance， continuity distance 共に根拠を得る事になり，また，計算量の軽減にもなると思われる．

3.6 VQ アルゴリズムの改良 その 1

3.6.1 プログラムのソースの抄訳

continuity distance の要素の一つとして melcepstrum を VQ 化しその量子間距離を計るものがある．この量子間距離は，前後の音素の関連性を考慮し，それぞれ重みを付けて加算する事で得られる。

```
static float nus_vector_dist(結合音素, 結合先音素);
{
    /* Find distance at join using vq (+) information.  May change */
    /* offset positions for join points                            */

    if (結合音素の前音素がポーズなら)
return 0.9;
    else if (結合音素と結合先音素が文脈中で連続しているなら){
return 0;
    else if (結合音素の前の音素と結合先音素の音が異なるなら){
return 1.0 + nus_vq_dist(p_unit,unit,cand); /* phones don't match */
    else if (結合音素が子音ならば)
return 0.1;
    else
return nus_vq_dist(p_unit,unit,cand);
}
```

3.6.2 解説と問題点

もし結合音素が子音であった場合に，無条件で接続しやすいとする定数を返しているが，異なるcontext間で結合する場合には，VQを求めるルーチンで最適な結合位置を求めておかないと，ラベリングが不正確で前の音素の最後を含んでしまっている場合などに，ノイズが入る要因となる．

3.6.3 対策

結合音素が子音の場合にもVQで望ましい結合位置を求めることにした。

具体的には、

```
    else if（結合音素が子音ならば）

return 0.1;
```

の行を削除した。

3.7 VQ アルゴリズムの改良 その 2

3.7.1 プログラムのソースの抄訳

VQ 距離を求める場合には，接続元 unit 終了部付近と接続 unit 開始部付近に window を設け，その window 内で最も VQ 距離の小さい部分を検索，両 unit の終了部・開始部を補正し，それを両 unit の VQ 距離としている．(Figure:3.3)

```
static float nus_vq_dist(int p_unit, int unit, NUS_cand_instance cand)

{

    /* Find the an acoustic distance between these units */

    /* Looks for good places to join near the ends of these units  */

    /* giving a score, uses vector quantization tables (built from */

    /* whatever -- cep deshou)                                     */

    /* The set up for window of frames to set is complex and       */

    /* therefore still full of bugs.  I've tried to rewrite it a  */

    /* number of times but its still gets messy                    */


    c_st = 結合音素側の結合部の window の先頭

    c_ed = 結合音素側の結合部の window の末尾

    p_st = 結合先音素側の結合部の window の先頭

    p_ed = 結合先音素側の結合部の window の末尾



    for (cs = c_st; cs < c_ed; cs++)

    {
```

```
        int incr = cs - cand_start_win_st;
        for (pe = p_st + incr;
             (pe < p_st + incr + 1) &&
             (pe < p_ed);
             pe++)
        {
            score = cs と pe の位置での
                    pitch,power,VQ の差の重みづけ後の総和
        }
        if (score < best_score)
        {
            best_pe = pe;
            best_cs = cs;
            best_score = score;
        }
    }
}

/* normalise score to 0.0->1.0 (full match is not 0.0 as only */
/* no join is 0.0  */
/* Looked at a distribution of vq scores and decided on the 40 above */
/* and 20 divisor below                                              */
fscore = 0.2 + (best_score/(10 + nus_vq_offset));
```

```
    return fscore;

}
```

3.7.2 解説と問題点

このプログラムでは，peはcsとcs+1の2値しか取らない。

durationが変わらないという利点はあるが，現時点でdurationはそれほど精密さを要求するパラメータではない。

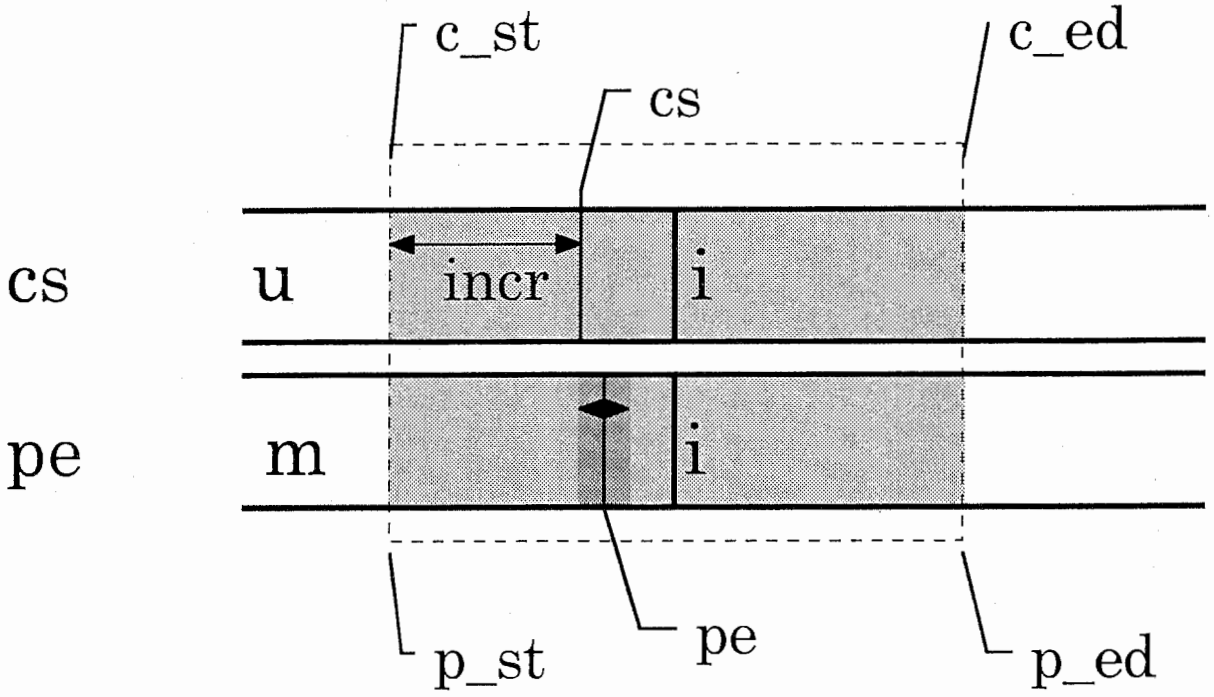


Figure 3.3: VQ最適接続部検索の概念図

3.7.3 対策と結果

durationと速度を犠牲にして，全pe,cs組み合わせについて最適結合部を検索する方式に変更した.

これにより，より最適な結合が導かれる可能性が増加した.

具体的には、

```
for (pe = p_st ;
     (pe < p_ed);
     pe++)
```

とした。

Chapter 4

評価

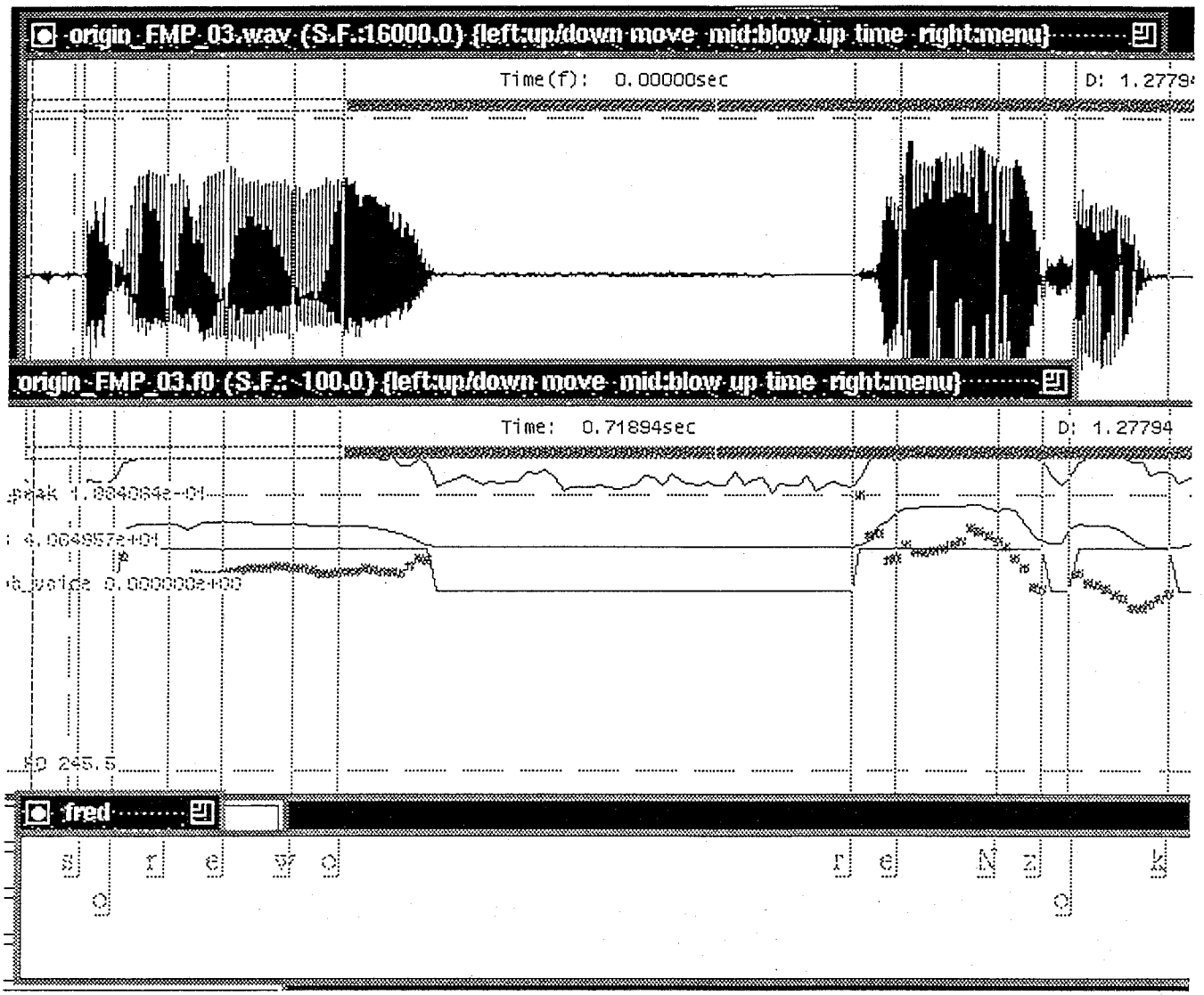
以上，CHATRの改良しうる点を述べてきたが，全てがすぐに最適解に影響する性質の物ではない為に，必ずしも合成結果に反映されるとは限らない.

合成音声の評価は元来非常に難しいため、いくつか合成波形およびf0、選択音素を図示するにとどめる。

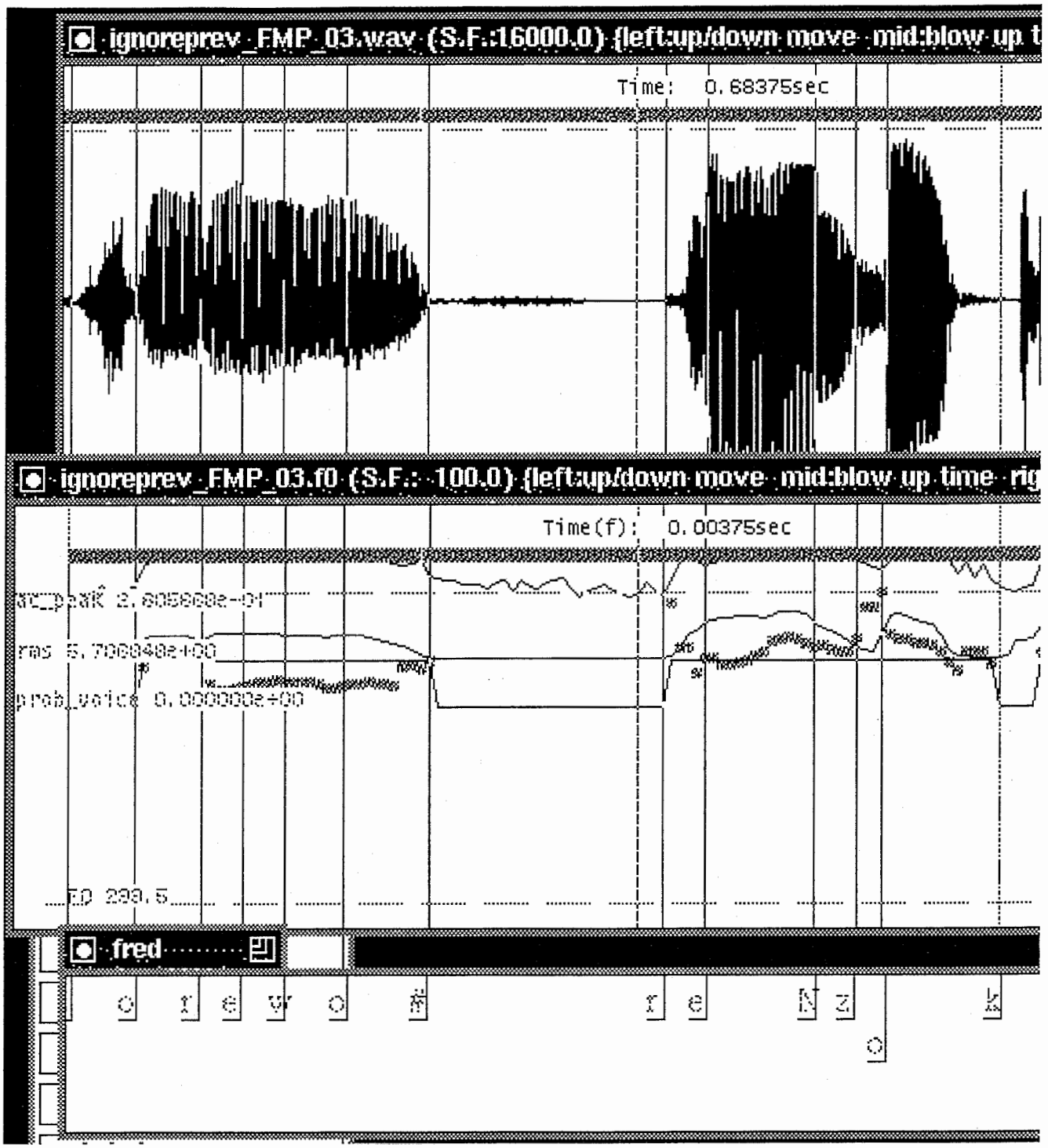
4.1 話者 FMP「それを、連続」

「それを」の後の duration,「連続」の韻率の再現性に向上が見られる。

4.1.1 改良前



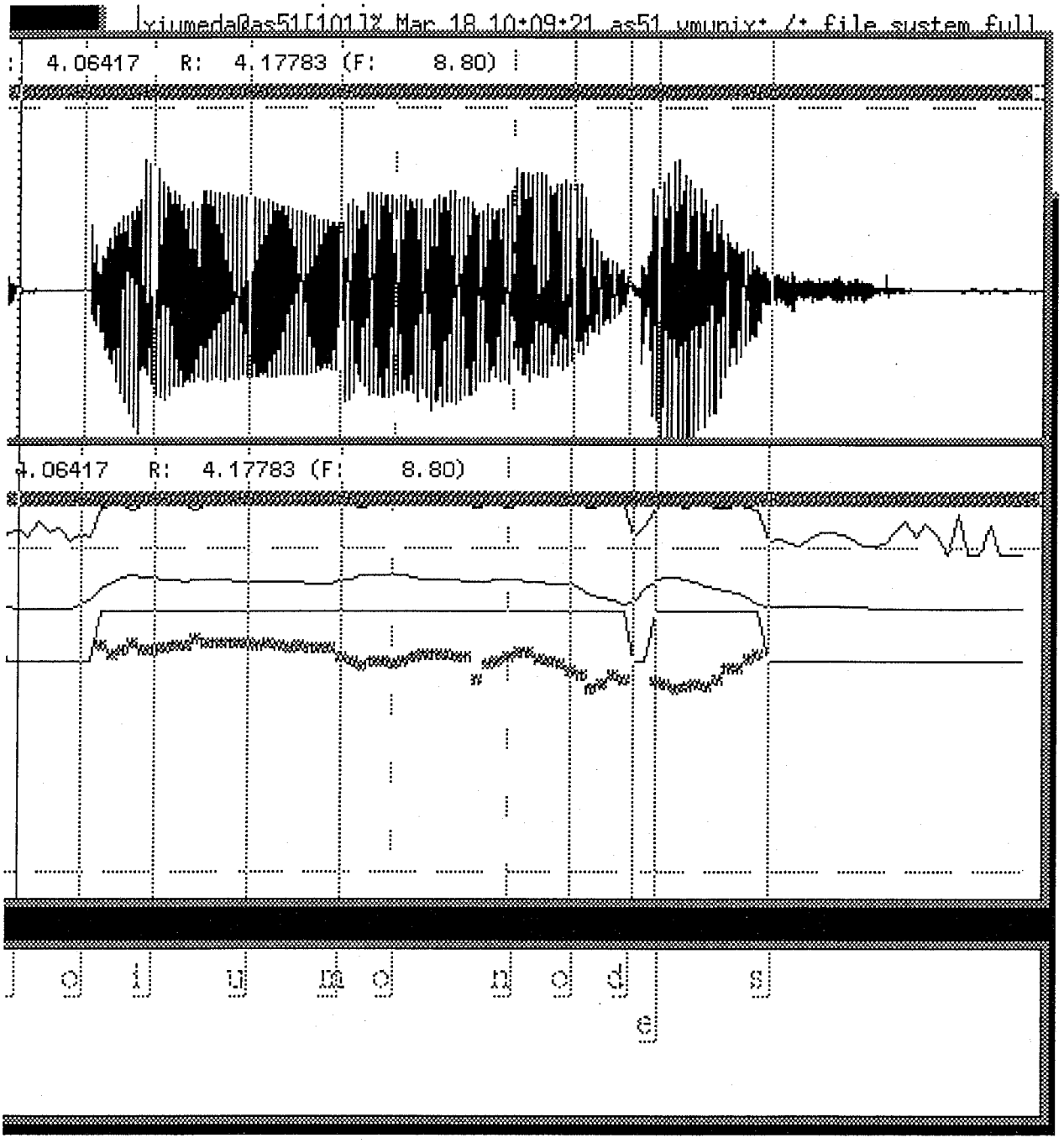
4.1.2 改良後


ignoreprev_FMP_03.wav (S.F.:16000.0) {left:up/down move mid:blow up t
Time: 0.68375sec
ignoreprev_FMP_03.f0 (S.F.: 100.0) {left:up/down move mid:blow up time rig
Time(f): 0.00375sec
rms 5.7000482+00
prob_voice 0.0000002+00
fred

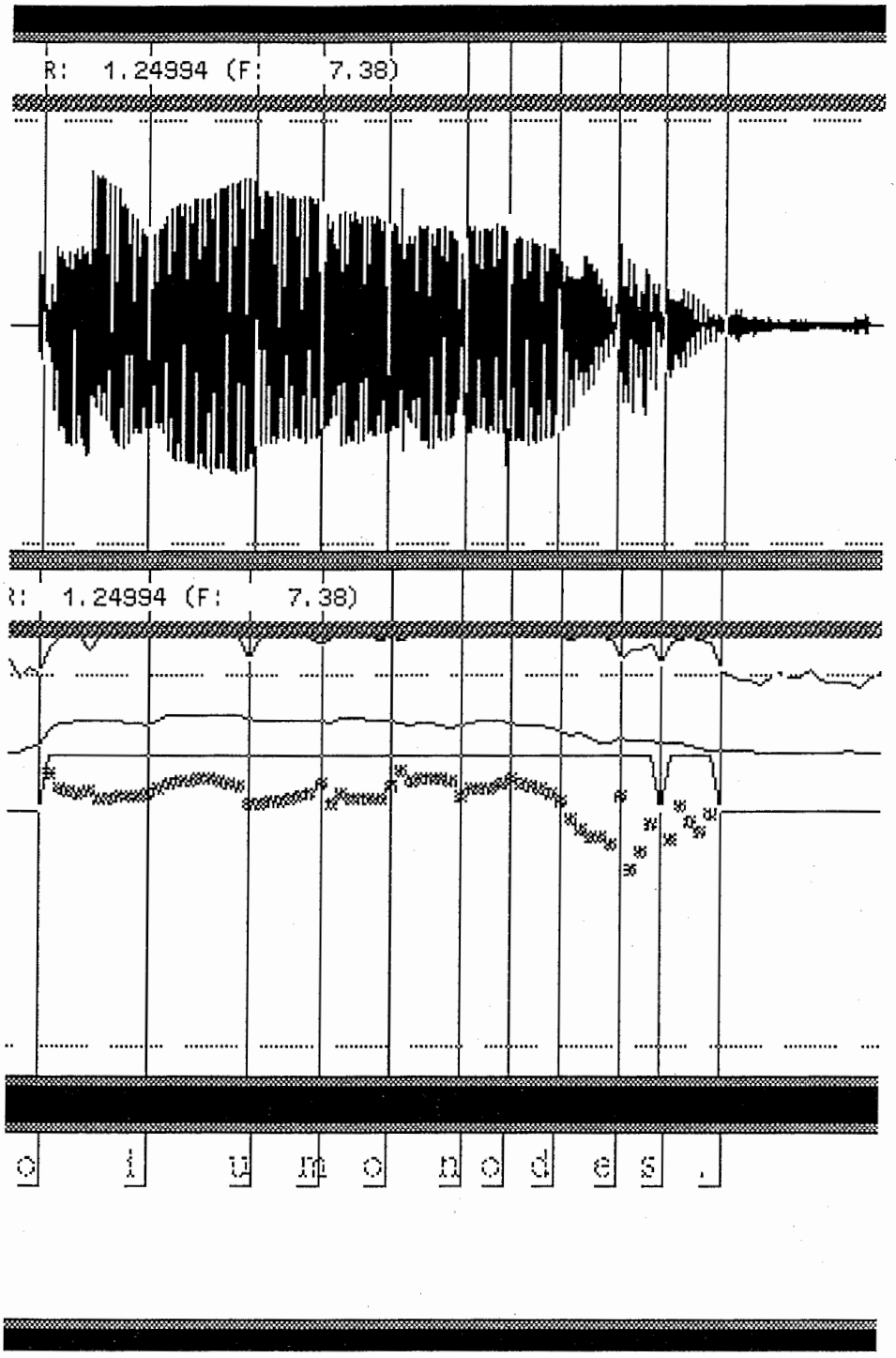
4.2 話者 FMP「というものです」

「ものです」のの韻率の再現性に向上が見られる。

4.2.1 改良前



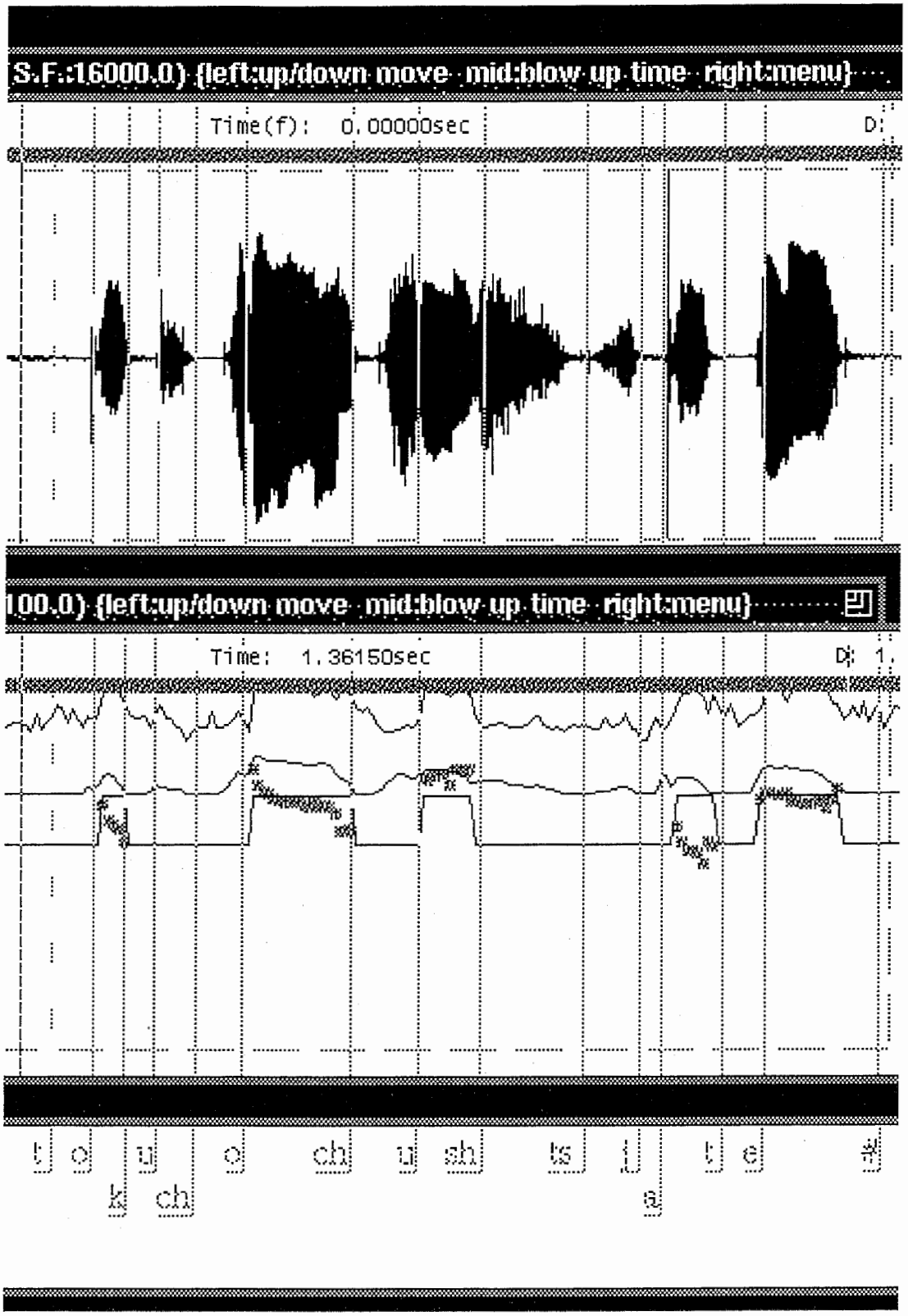
4.2.2 改良後



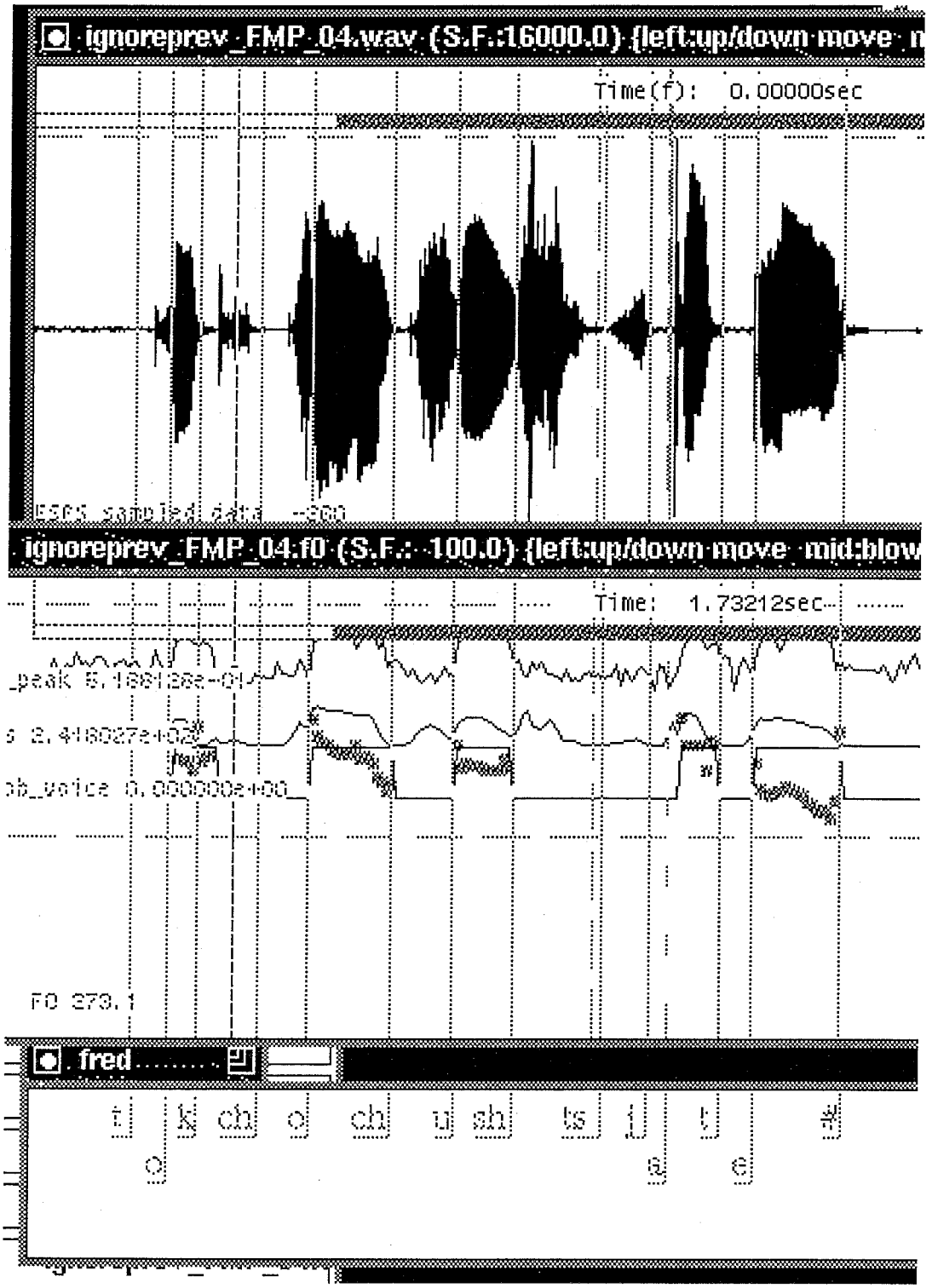
4.3 話者 FMP 「特徴抽出過程」

「過程」の韻律の再現性に向上が見られる。

4.3.1 改良前


S.F.:16000.0) {left:up/down move mid:blow up time right:menu}
Time(f): 0.00000sec
D:
100.0) {left:up/down move mid:blow up time right:menu}
Time: 1.36150sec
D: 1.
t o u o ch u sh ts i t e
k ch
a

4.3.2 改良後


ignoreprev_FMP_04.wav (S.F.:16000.0) {left:up/down move
Time(f): 0.00000sec
ignoreprev_FMP_04.f0 (S.F.: 100.0) {left:up/down move mid:blow
Time: 1.73212sec
FO 273.1
fred
t
k
ch
o
ch
u
sh
ts
t
o
a
e

Chapter 5

まとめと今後

5.1 まとめ

CHATR のアルゴリズムの改良として実際に行なったことは、

- VQ による最適接続部を求めるルーチンに訂正を行ない、接続性を上昇させた。
- continuity dystance を melcepstrum の VQ と cepstrum により求め、音声単位候補選択時の context 及び duration への依存度を下げた。

これらにより、

- 子音、母音の duration の再現性が向上した
- 韻律の再現性が向上した

という結果を得た。

5.2 今後

これまでに述べた以外に今後 CHATR に残る課題として，以下のような物があると思われる.

- 両 distance の構成要素間の相関を求める事で，構成要素間の重み付け係数をより統計的に定義することが可能だと思われる．また，重み付け係数を定数ではなく関数で求める事が可能ならば，データベースからの自動学習による重み付け係数決定が可能であるかもしれない.
- 音素間の音響的距離に母音や子音，及びそれらの特徴量の概念を導入する事で，より音響的特徴に密着した音素選択が可能になる．これは context による両 distance 推定に音響的根拠を与え，また，データの圧縮や演算量の削減等に役立つと思われる.

- データベースの最小単位を現在の音素ではなくVQのframeとし，入力音素列の1音素を複数の音素の切りだしから合成する事を可能にすれば，特に伸ばす音の表現等での表現力の飛躍的な向上が可能だと思われる.

今後CHATRに携わる皆様の活躍を期待します:D

Bibliography

[1] Nick Campbell : "韻律を用いた音声合成の単位選択", 日本音響学会平成 6 年度秋期研究発表会講演論文集 , 2-5-10 , 285-286(1994)

[2] Abe K. , Takeda K. , Sagisaka Y . (1989) : "On the concatenation of speech synthesis units according to unit extraction context" , IEICE JAPAN , SP89-66,17-22

[3] 佐藤俊則 , "今日からあなたも chatr ユーザー" ,
http://www.itl.atr.co.jp/local_info/department/dept2/dvi/jchatr.dvi,
April 8th 1996

[4] Martyn Weeks , "CHATR – a Generic Speech Synthesis System" ,
http://www.itl.atr.co.jp/ mweeks/chatr_toc.html ,
February 25th 1997

Chapter 6

謝辞

本研究の遂行にあたり，アドバイザーの樋口宜男室長，直接指導者の Nick Campbell 主幹研究員，数々のご指導を賜った佐藤俊則様・西村政宏様・ Wen ding 様 他第二研究室の皆様， Final talk にてご指導を賜った匂坂芳典一研室長，他ＡＴＲ音声翻訳通信研究所の皆様に感謝します．

ＡＴＲへの学外実習の機会を下さった白井克彦早大理工教授に感謝します．及び早大理工白井研の皆様に感謝します．

同じく白井研からＡＴＲへの学外研修生であり友人の星野君に感謝します．

父に，ありがとう．母に，さようなら．

全ての子供達に「おめでとう」