人工喉頭を用いたSinger Robotの研究ー母音フォルマントのリアルタイム調整ー

中野, 陽介 / NAKANO, Yousuke

(発行年 / Year)
2008-03-24
(学位授与年月日 / Date of Granted)
2008-03-24
(学位名 / Degree Name)
修士(工学)
(学位授与機関 / Degree Grantor)
法政大学 (Hosei University)

2007 年度　修士論文

人工喉頭を用いた Singer Robot の研究

― 母音フォルマントのリアルタイム調整 ―

Study on Singer Robot Using Artificial Larynx

- Real-time Adjustment of Vowel Formant -

指導教員　高島　俊　教授

法政大学大学院工学研究科

機械工学専攻修士課程

06R1129　　中野　陽介

目次

第1章 序論

1.1 ロボットの歴史および研究背景

「ロボット」という言葉は 1920 年にチェコの作家カレル・チャペックによって書かれた，戯曲「R.U.R」で最初に使われたのとされ，その語源はチェコ語で労働の意味である「robota」と考えられている．ロボットの定義は現在も議論が続いているが，仮に「人間型の人工物」と考えるならば，その概念は非常に古い．紀元前 3 世紀ごろの作品とされる「アルゴ探検記」には「青銅人間タロス」と呼ばれる人型の機械が書かれている．また，ユダヤ教の伝承にも泥で作られたロボットであるゴーレムが登場する．このように，人間は古くから人間型のものを作ることに興味を持っていた．

機械の技術が進化し，18 世紀にはいると人間や動物の動きを真似た自動人形，いわゆるからくりが作られた．有名な自動人形にヴォーカンソンのアヒル(1738)が挙げられる．（参考文献[4]）日本でも，1662 年に竹田近江が自動人形による芝居「竹田からくり座」を行なったと記録が残っている．その他，茶運人形などのからくり人形が江戸時代に作られた．

ロボットは機械工学のほか，電気電子工学，制御工学など様々な分野にわたるが，ロボットの開発が大きく進歩したのはコンピュータの発展によるところが大きい．フォン・ノイマンによって考案されたプログラム内蔵方式のコンピュータは現在のコンピュータの基本となっている．

ロボットはまず産業用のものが多く開発された．特にロボットが積極的に導入されたのが製造業の分野である．その工場の現場において，労働力不足を解消するだけでなく，人間にとって危険な作業を代わりに行う，作業の速さと正確さを向上させる，といったことに大きく貢献した．加工機械としては，1952 年パーソンズと MIT が協力して NC(Numerical Control)工作機械の開発を成功させた．（参考文献[4]）この他にも，組立てロボット，溶接ロボットなどが開発されている．このように，産業用ロボットは，その需要の大きさから早くから研究開発が行われ，発展してきた．

一方，産業用ロボットのような実用的なロボットとは違う目的・方向性のロボットとして，エンターテイメント・アミューズメントロボットの研究開発が近年多くなりつつある．エンターテイメント・アミューズメントロボットの役割は，その名の通り，「人間を楽しませる，魅了する」ことである．そのため，ロボットを作製するに当たって，どのようにすればより人間を楽しませ，魅了

することができるかを常に考慮に入れなければならない．また産業用ロボットと比べると，ロボットの専門家ではない一般人と距離感が近いため，その安全性の追及も不可欠となる．現在までにアミューズメント用として開発されてきたロボットには様々な種類がある．例えば，テーマパークで見られるような，動物や植物の形状を模したロボットや，楽器を演奏するロボット，二足歩行をするロボットなどがある．

二足歩行をするロボットに関しては，本田技研工業の ASIMO がよく知られている．ASIMO は人間すでに人間社会の中で簡単な会場案内，説明，受付などを行っている．さらに，音声認識技術や画像処理技術を用いて，対象者を判断するなどのコミュニケーション機能を持っている．

また，ソニーは 1999 年にペットロボットの AIBO を世界で始めて発売した．AIBO は動物らしい動きをするためのアクチュエータを持つだけでなく，様々なセンサを持ち，多くの情報を取り込み学習することができる．そして自分で考えて行動を起こし，あらかじめプログラムされたものであるがいくつかの感情を表現することができる．このように，ロボットとコミュニケーションを取って楽しむことができる機能を持ったアミューズメント用ロボットが開発されている．

本研究室では，これまでに楽器を自動演奏するロボットや人の運動を模倣するロボットの研究をおこなってきた．自動演奏ロボットとして，サクソフォン演奏ロボット，トロンボーン演奏ロボット，トランペット演奏ロボット，尺八演奏ロボットがある．運動ロボットとして，鉄棒ロボット，トランポリンロボット，マウンテンバイクロボット，投打ロボットがある．

1.2 研究目的

「歌を歌い、聞かせる」という行為は、人が道具を使わずに表現できる芸術活動の一つである．その行為は，人の心を癒すことや感動させることができ，世界中で親しまれている．

ロボットの世界でも人を感動させたり，楽しませたりすることを目的としてアミューズメントロボットが研究されている．では，ロボットも人と同じように歌うことによって人の心を癒し，感動させることはできないのだろうか．現在，ロボットと人間とのコミュニケーション手段として音声を用いるものや，人間の音声や聴覚をロボットにより表現する事でその仕組みを解明しようとする研究も行われてきている．既存の各種ロボットにおいては，音声合成と音声認識による会話を目指すものが多い．しかし，音声合成によるスピーカーからの発声音はまだ機械的に感じられるだろう．この声は情報の伝達においては役割を果たすものの，歌うロボットとしてのアミューズメントロボットには違和感がある．歌うという行為には，人が発声する原理と同じように，声帯による発声の方が適しているのではないだろうか．

本研究の目的は、合成音を用いずに人工声帯を用いた発声によって「歌う」ロボットを実現させ，かつ，音の高さの制御も行うことにより，感情表現も可能な「歌う」ロボットを作るというこれまでにないものである．最終的な到達点としては，楽器を演奏するアミューズメントロボットだけで構成されたバンドのボーカルとして歌うことである．

1.3 本研究の概要・特徴

本研究は人の歌う行為を模倣し，人工喉頭を用いて「歌う」ロボットの研究開発である．「歌う」行為は,「言葉を発すること」と「メロディー・リズムを表現すること」に分けることができる．本研究でも，この 2 つをロボットで実現することを目指している．これまでの研究により、母音の連続生成とピッチ変化が可能なロボットが開発された。それに加え、さらに子音の発声を可能にする機構の開発や、ビブラート等により感情を表現する方法などについての研究が必要となるだろう．本論文では，その歌うために必要な要素の中でも，言葉の一部である母音に焦点を当てている．

「歌う」には，明瞭な母音の発音によって言葉を生成することが不可欠だろう．そのためには，各母音が「フォルマント」という音響的特長を正確に持つ必要がある．ロボットが発声した母音のフォルマントも自由に調整できるようにすれば，歌う行為に適した発声が可能となるだろう．そこで，本報告ではロボットが発生した母音がより的確なフォルマントを持つように，フォルマントを調整するシステムを開発した．そのシステムでは，ロボットの声からフォルマントを抽出し，ロボット自身にフィードバックさせることでフォルマントを再調整する，いわゆる聴覚フィードバックをおこなう．本研究では，このシステムをリアルタイムフォルマント調整システムと呼んでいる．

特徴はロボットの声を一度 WAVE ファイルとして録音してからフォルマントを抽出しフィードバックさせるのではなく，リアルタイムでフォルマントを抽出しフィードバックさせる点である．これにより，ロボットが歌っている最中でも，フォルマントを最適な値に調整し明瞭な母音を発声することができる．

1.4 本論文の内容と構成

本論文では,「フォルマント調整プログラム」と「フォルマント抽出プログラム」という 2 つのプログラムを作成し，最終的にリアルタイムでのフォルマント調整を実現させた.

各章に内容と構成を説明する.

「第 2 章　発声のメカニズム」では人間の発声のメカニズムについて説明している．最初に発声に必要な原音発声器官と調音器官を紹介する．その次に，声について説明し，母音，子音，フォルマントについて述べている．本研究はフォルマントの調整についての研究であるため，よく確認してもらいたい.

「第 3 章　Singer Robot」では本研究で使用する Singer Robot を紹介する．開発するシステムは，この Singer Robot に対応するように作成している．ロボットの詳しい内容は，参考文献[1][3]に記載されている.

「第 4 章　リアルタイムフォルマント調整システム」では本研究で開発するシステムの概要を説明する.「フォルマント調整プログラム」と「フォルマント抽出プログラム」，ロボットのコントロールプログラム，Singer Robot のつながりを確認してほしい.

「第 5 章　フォルマント調整」では調整ベクトルという考え方からフォルマントの調整アルゴリズムを考案する．本研究の心臓部といえる項目である.

「第 6 章　フォルマント抽出」では音声データからフォルマントを抽出する方法について記述する．フォルマントの抽出には線形予測法を用いている．また，音声データを加工する際の前処理について紹介している.

「第 7 章　ロボットとのリンク」では第 5 章，第 6 章で作成した 2 つのプログラムをロボットに繋げる作業をおこなう．プログラムの各パラメータの単位の変換などをする．最終的に完成したアプリケーションについても紹介している.

第2章 発声のメカニズム

2.1 人間の発声器官

ロボットで歌う行為を再現するためは，人の発声のメカニズムについて知る必要がある．まず人の発声器官について説明する．

音声の生成は人間の発声器官が運動することによっておこなわれる．その過程は次の通りである．

1) 呼吸運動（肺から空気を圧力をかけて押し出す）
2) 声帯振動（声帯が振動し，原音を発声する）
3) 共鳴（原音が喉や口腔，鼻腔で共鳴する）
4) 調音（共鳴された音を認識できる母音，子音に変える）
5) 放射（音声を口腔や鼻腔から大気に放出する）

これらの過程は原音発声器官(肺，声帯)と，調音器官(口腔，鼻腔，舌，歯，唇)の 2 つの系に分けることができる．それぞれの機能や構造に述べていく．

2.1.1 原音発声器官

原音発声器官は，肺と喉頭（声帯）からなる．肺では，空気の吸入，排出がおこなわれるが，人間が発声する際には呼気を利用して喉頭の中にある声帯を振動させて原音を生成する． 喉頭の断面図を Fig.2.1 に示す．この断面図は，正面から喉頭を見た図である．肺から押し出される呼気は，気管の内の 3 つの弁を通過することになる．声帯はそのうちの 1 つであり，Fig.2.1 に示すように一番下についている．他の 2 つは，それぞれ喉頭蓋，仮声帯と呼ぶ．喉頭蓋は呼気を完全に止める作用を持つが、仮声帯の役割はまだ正確に解明されていない．声帯は，呼吸時には大きく開いているが，発声する際には隙間が狭まる．この間を空気が通り抜けるとき，ベルヌーイ効果によって声帯が閉じる．閉じた声帯は，肺から送られてくる空気の圧力で再び開く．これらを繰り返すことにより，空気の周期的な断続が起こり，原音を発生させることができる．これを声帯音源（glottal source）と呼ばれ，およそ－12dB/oct の音である（参考文献[7]）．声帯筋を緊張させると、声帯の開閉の周波数が高くなり、声が高くなる．また，呼気流を大きくすると大きな声となる．

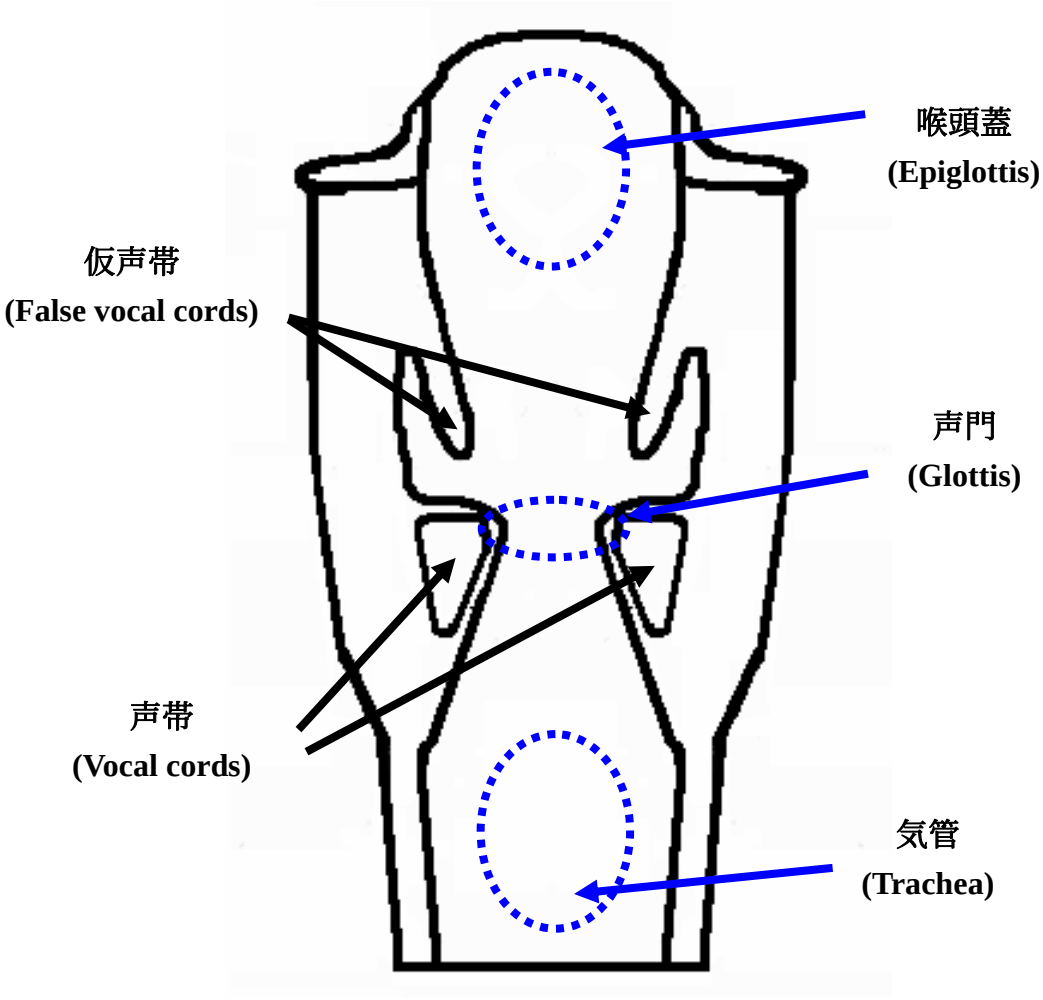

Fig.2.1　喉頭の断面図

2.1.2 調音器官

人間の調音器官は唇，歯，そして舌からなる口腔と，鼻腔とからなる．調音機構の構造を Fig.2.2 に示す．人間の調音行動は，顎，舌，そして唇などを動かして声道の形状を特定の形状に変化させ，ある周波数特性を持った音響管を作ることにあたる．原音発声器官で生成された原音が声道を通過することで，ある周波数を持つ音波が強められ、ある周波数の音波は弱められるという共鳴現象が生じ，原音が調音され声として聞こえてくるのである．

声道の形状は，母音の発声時にはほぼ変化しないが，子音を発声するときには，舌や唇で声道を一瞬塞いだり，声道を狭め乱流雑音を生成したり，各部位が複雑な動きをする．特に，舌は自由に形を変えることができ，また運動させることができるので，口腔内の形や容積を変化させ，母音をはじめ多くの調音に重要な役割を果している．（参考文献[8]）また，鼻腔は口腔とは違い，形状が常に一定であるため，得られる共鳴特性も一定である．鼻腔が音声の生成に影響するのは，子音の中の鼻音[m]，[n]を発声するときである．このとき，鼻腔と口腔を繋ぐ軟口蓋が開閉することによって，鼻腔を利用した音声を生成する．母音や他の子音を発声するときは軟口蓋が閉まり，鼻腔と口腔は繋がっていない．

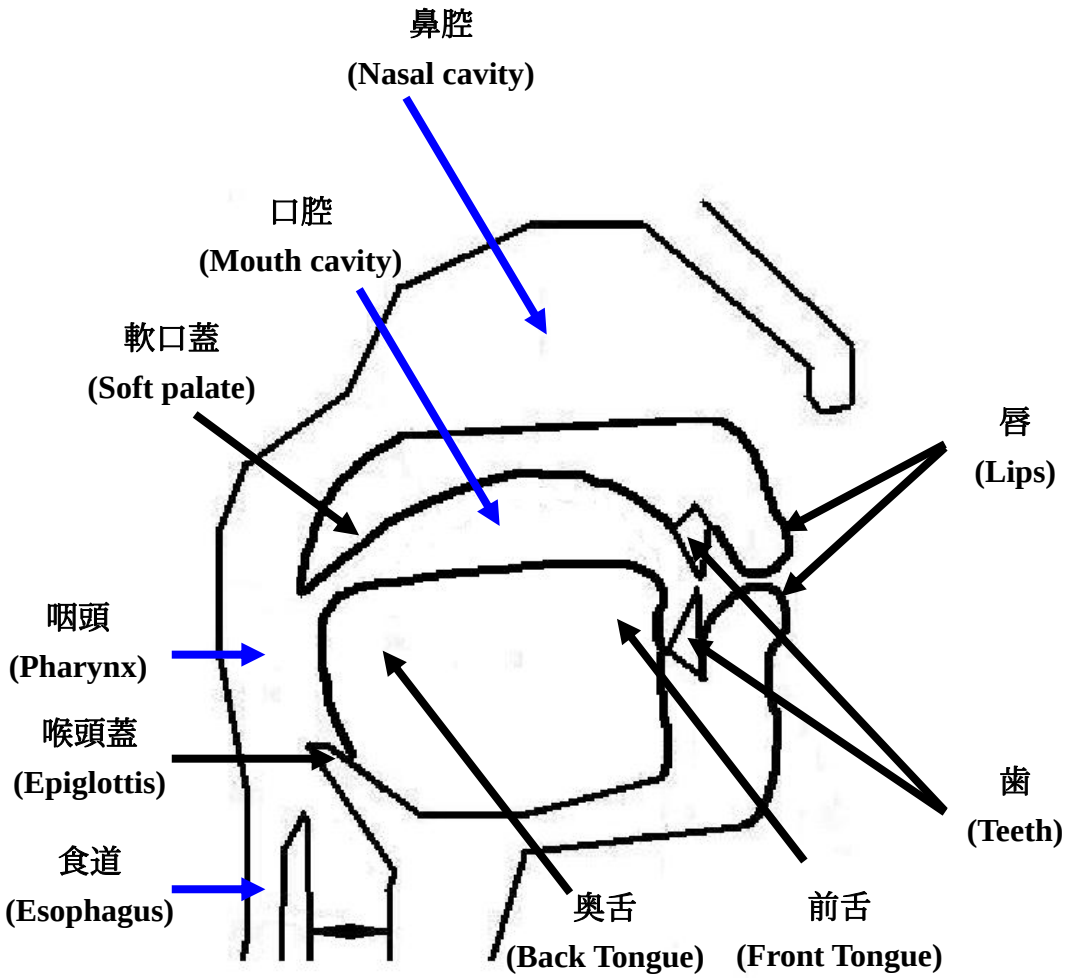

Fig2.2　調音器官の断面図

2.2 声の種類

2.2.1 母音と子音

歌うロボットを開発するためには，次に人の声について知っておく必要がある．言葉は母音と子音の組み合わせによって成り立っている．ここでは，母音と子音の特性について説明する．また，本研究で重要となるフォルマントについても述べる．

2.2.2 母音の分類とフォルマント

2.2.2.1 母音の分類

母音は，声帯を振動させて原音を生成する有声音である．この原音が声道を通過することにより調音がおこなわれる．声道の形状，断面積が変化することにより，声道の伝達特性が変化し，各母音のスペクトルが生成される．母音はその発声時に声道の形状が変化しない定常的な音である．

母音の分類を Fig2.3 に示す．この図は，母音と顎，舌の位置との関係を示したものである．横方向の位置は，左ほど声道の狭めが唇に近いことを示す．縦方向の位置は，下ほど顎が開いていることを示す．例えば，日本語母音[i]を発声するときの顎は，顎がほとんど閉じられており，下が唇近くに移動して声道の狭めが前に来ていることがわかる．このように声道の形状は主に舌の位置と顎の開き方で決まる．

千葉，梶山は日本語の母音の発声時の声道を X 線写真と内視鏡により計測した．その図を Fig2.4.1～Fig2.4.5 に示す．それぞれの母音の声道を正面と側面から見た図である．この図は，声道を正面から見た断面図と，側面から見た断面図からなっている．そして，声道をいくつかの要素に分割して，唇から－1，0，1…と番号をつけて断面積を掲載している．そして，その断面積と，声道を円筒とみなして算出した半径のグラフを掲載している．

また成人女性(20 代)が発音した 5 個の日本語母音の波形とスペクトルを Fig2.5.1～Fig2.5.5 に示す．音声の性質を解析する際に重要なことの 1 つとして，その音声の持つ周波数成分を知ることがある．特に，人間は母音を周波数成分によって識別している．特に母音はそのスペクトル包絡に注目する必要がある．

日本語の母音[a]，[i]，[u]，[e]，[o]の声道の形状の特徴を挙げる．

[a]の声道は，咽頭部の断面積が小さくなり，口腔部の断面積と開口部の断面積が大きくなっている．これは舌が後方に移動し，唇が開かれているためである．

[i]の声道は，咽頭部の断面積が大きくなり，口腔部，開口部の断面積は小さくなる．これは舌が前方に移動して口腔の側面に接触しているためである．

[u]の声道は，奥舌と軟口蓋とが接近して狭めを作り，声道がほぼ同じ大きさの咽頭部と口腔部に分けられる．

[e]の声道は, [i]の声道と同じように咽頭部の断面積が大きくなり, 口腔部, 開口部の断面積は小さくなるが，口腔部と開口部の断面積は[i]に比べて広くなっている．

[o]の声道は，[a]の声道と同じように咽頭部の断面積が小さくなり，口腔部の断面積大きくなっているが，開口部の断面積は小さくなっている．唇が閉じられているためである．

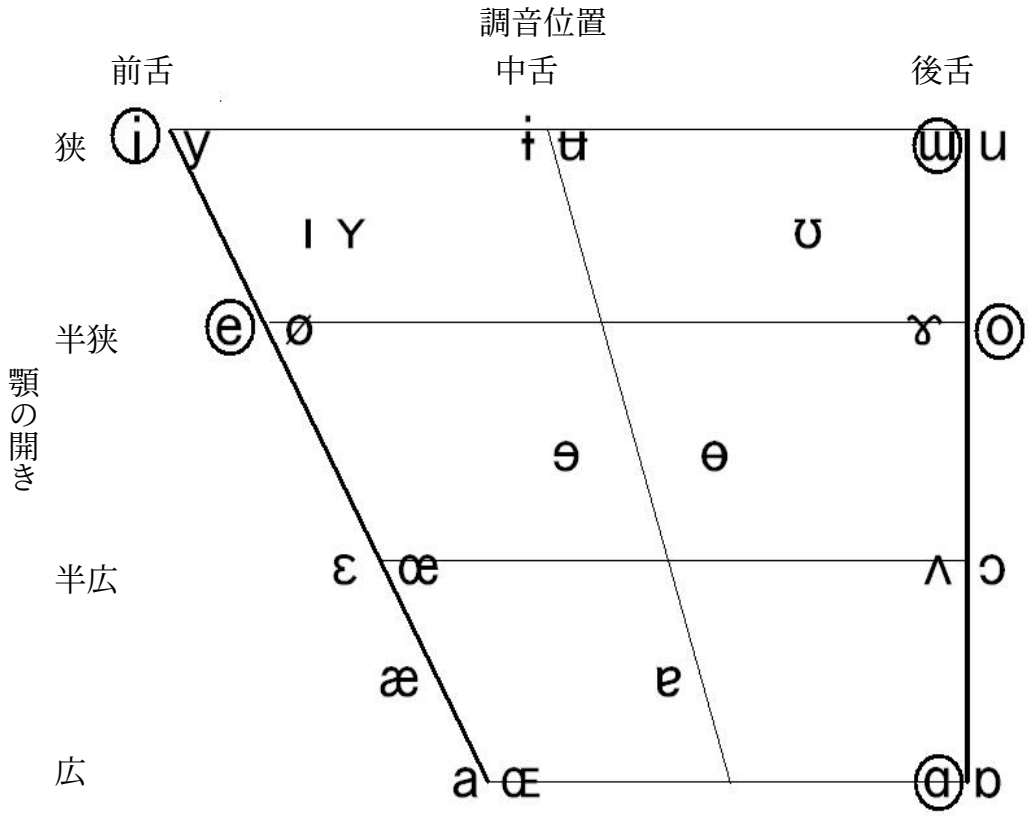

Fig2.3　母音の分類（〇は日本語母音）（参考文献[12]）

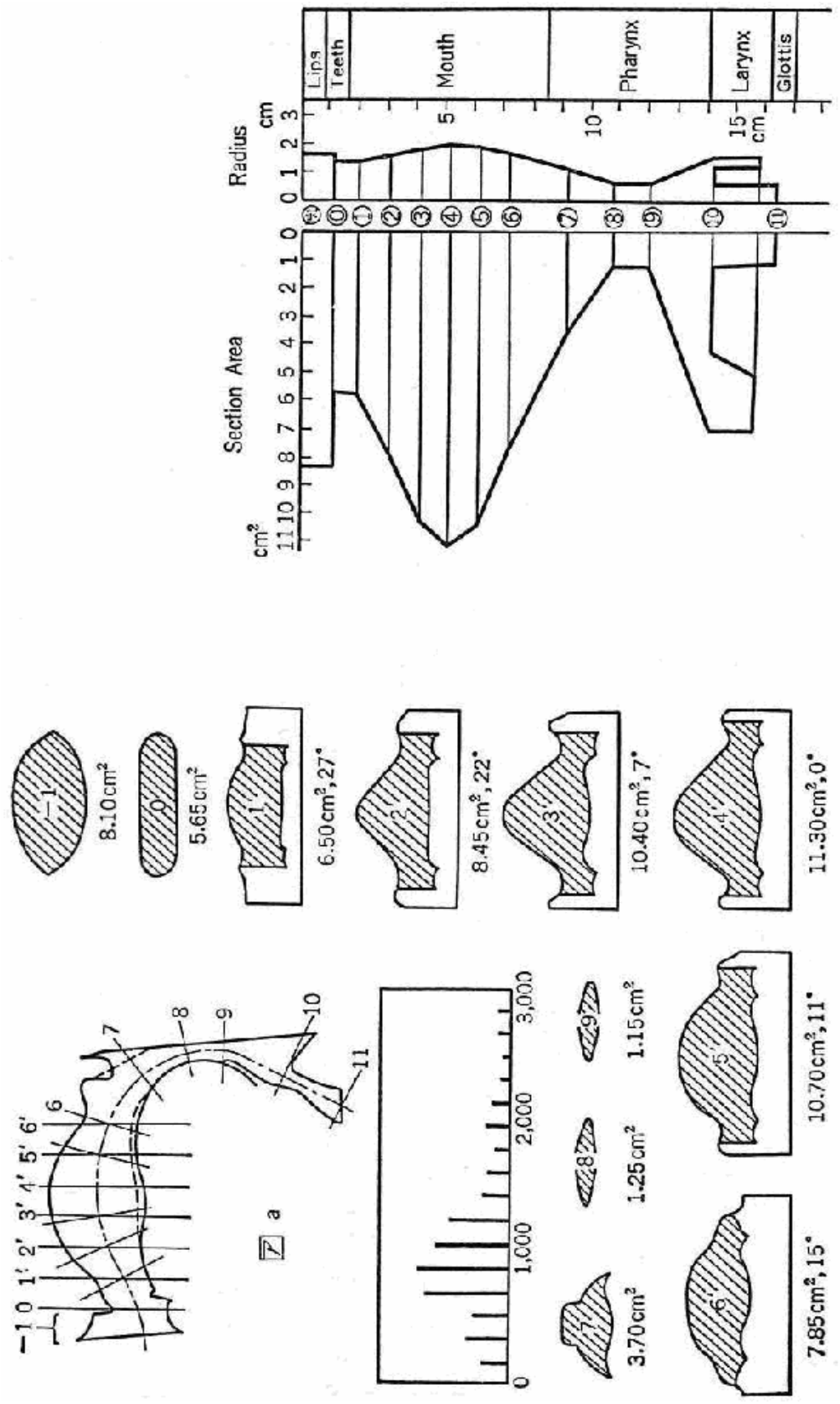

Fig2.4.1　[a]の声道断面積（参考文献[1]）

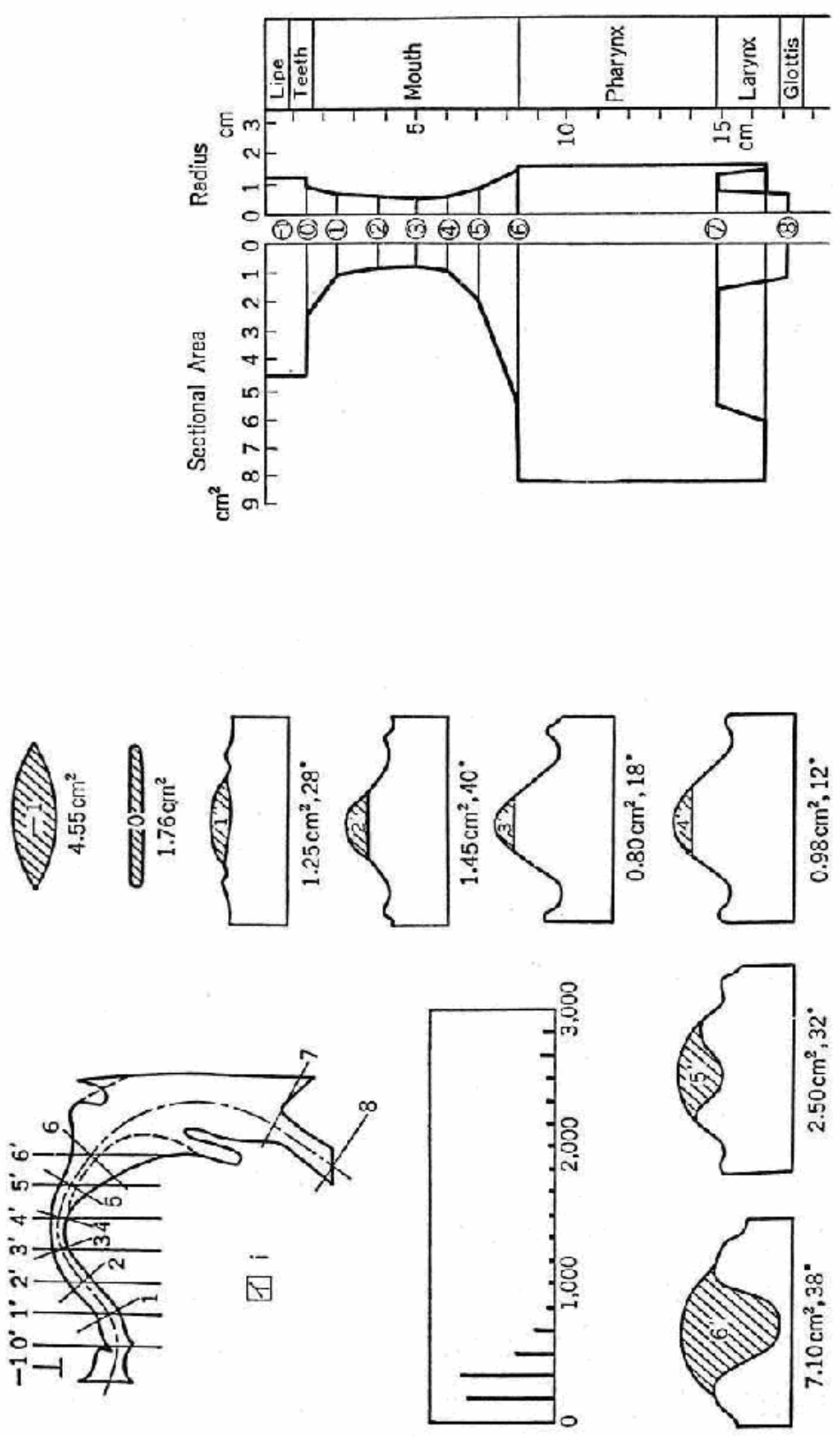

Fig2.4.2　[i]の声道断面積（参考文献[1]）

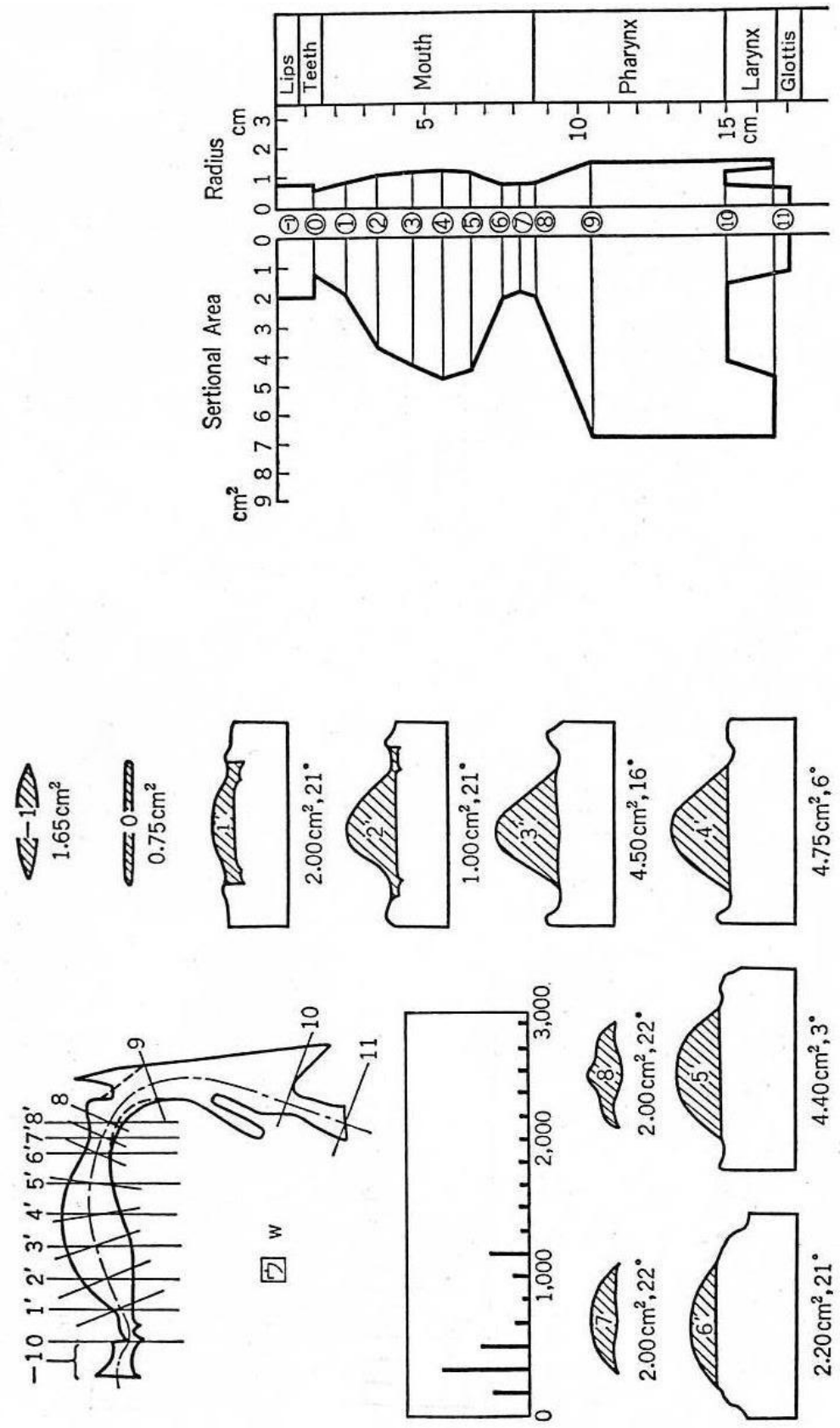

Fig2.4.3　[w (u)]の声道断面積（参考文献[1]）

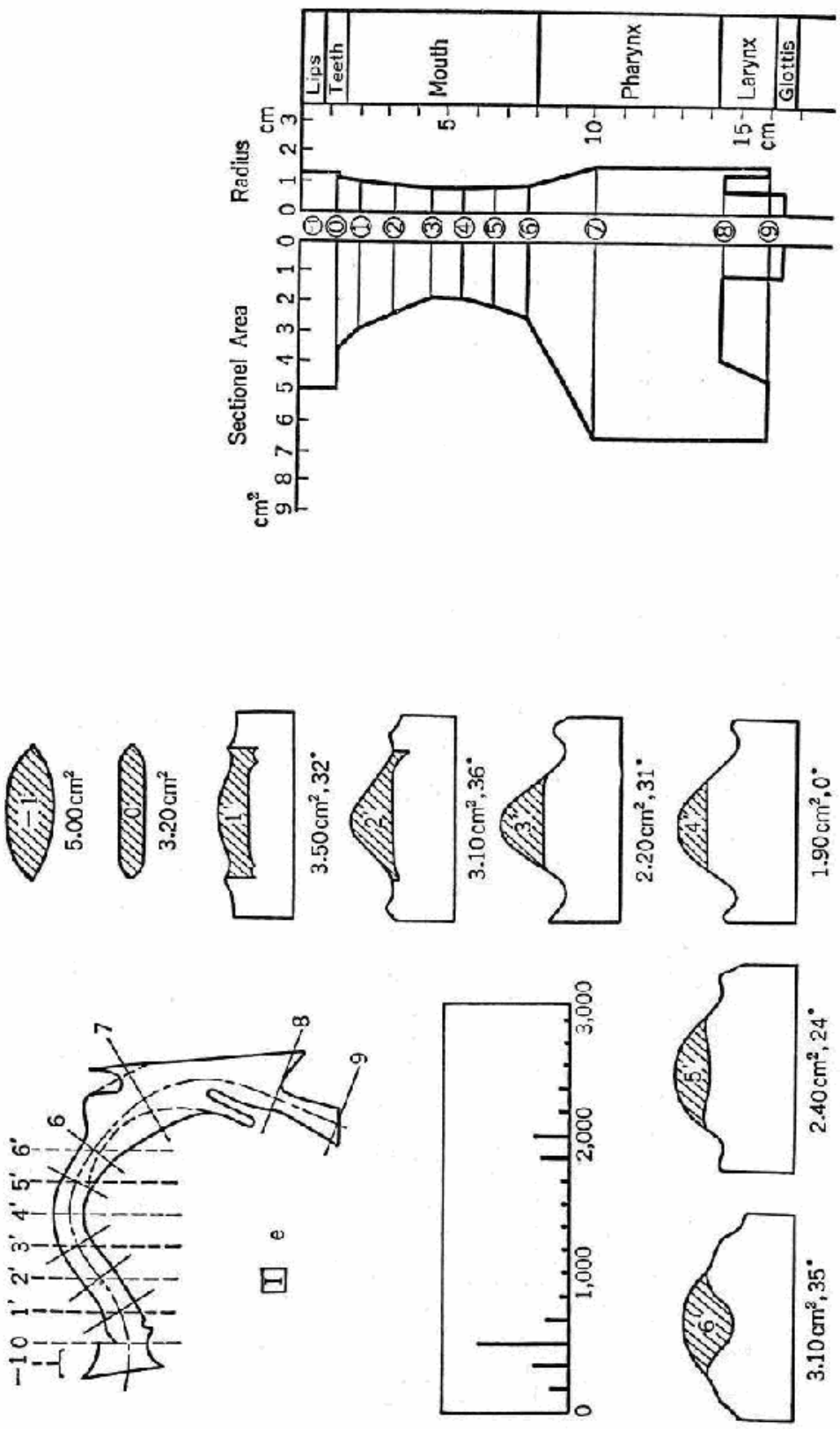

Fig2.4.4　[e]の声道断面積（参考文献[1]）

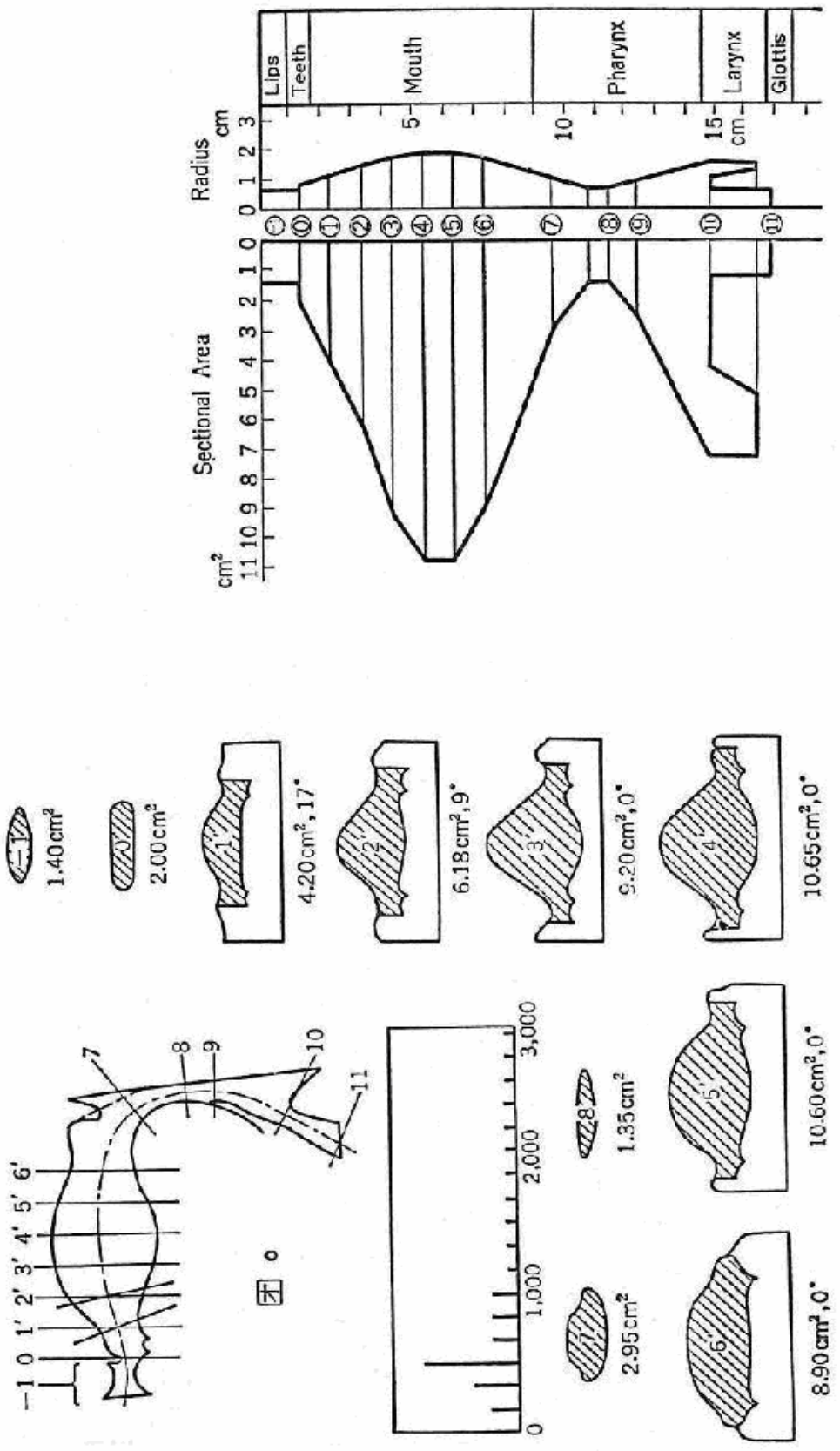

Fig2.4.5　[o]の声道断面積（参考文献[1]）

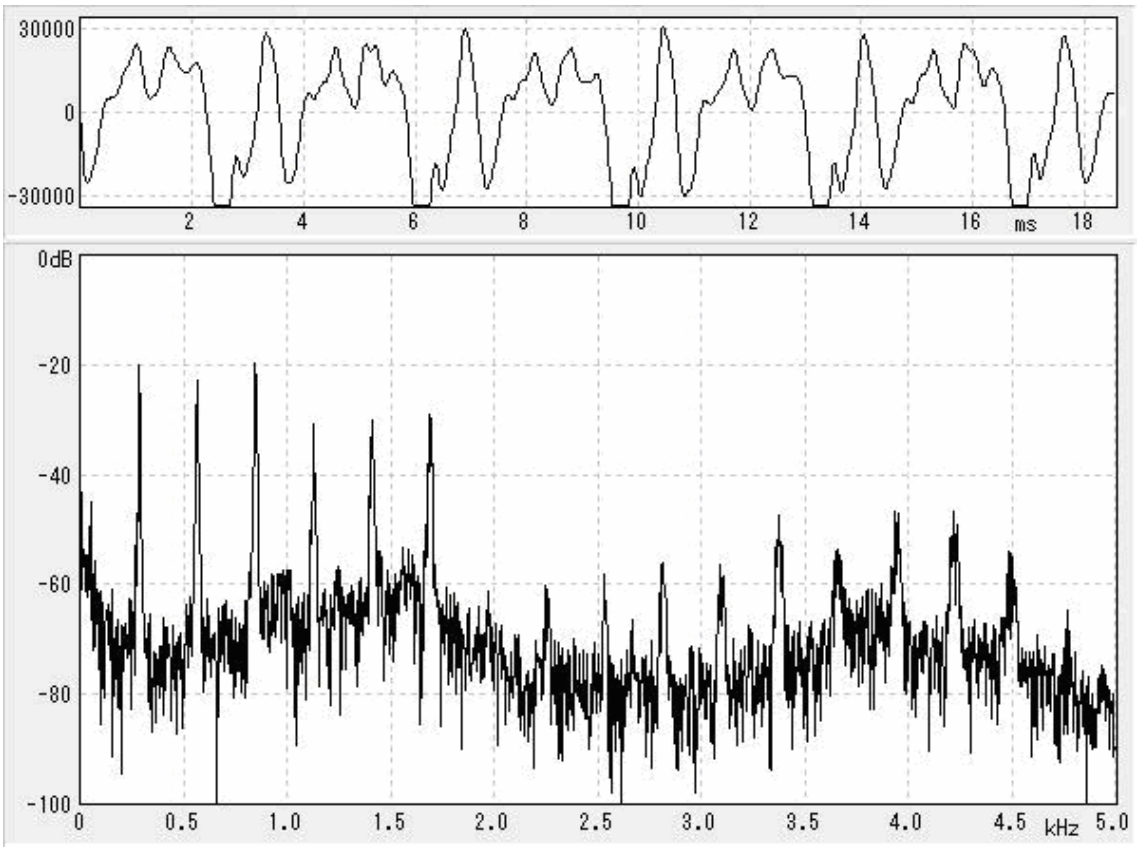

Fig2.5.1 日本語母音[a]の波形及びスペクトル

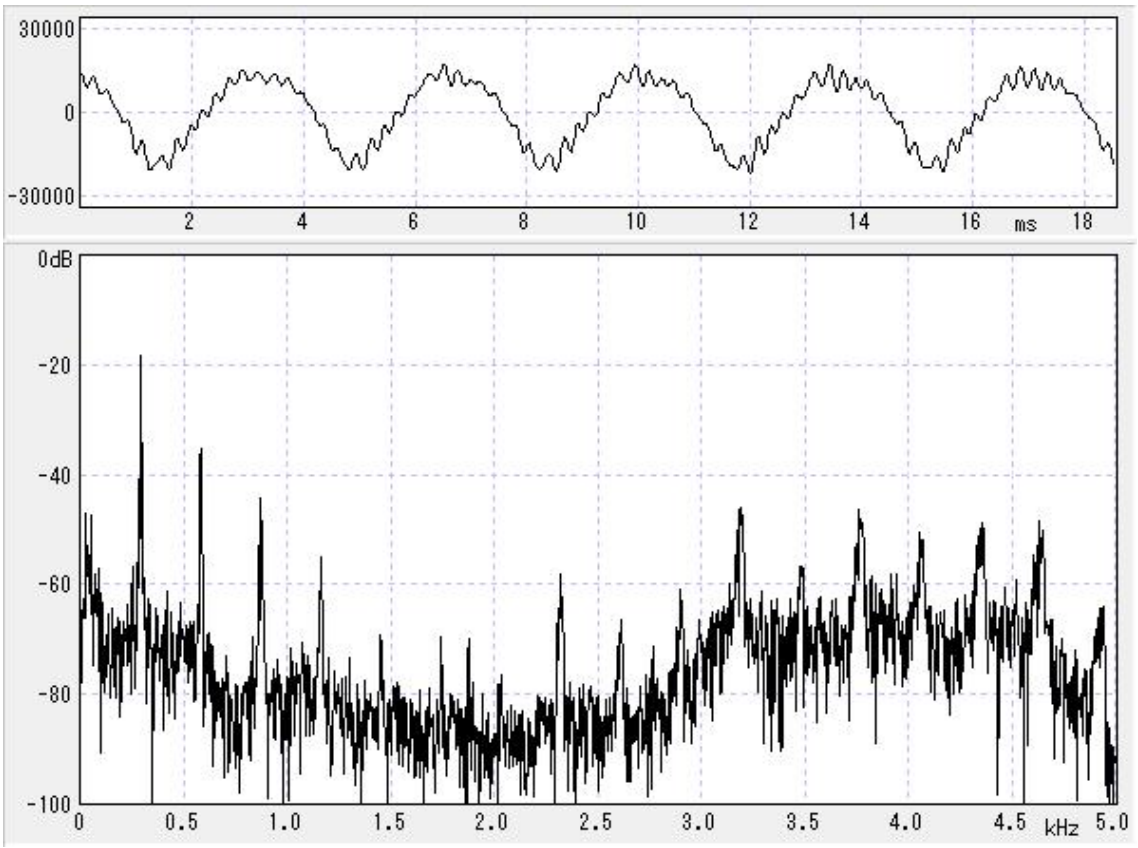

Fig2.5.2　日本語母音[i]の波形及びスペクトル

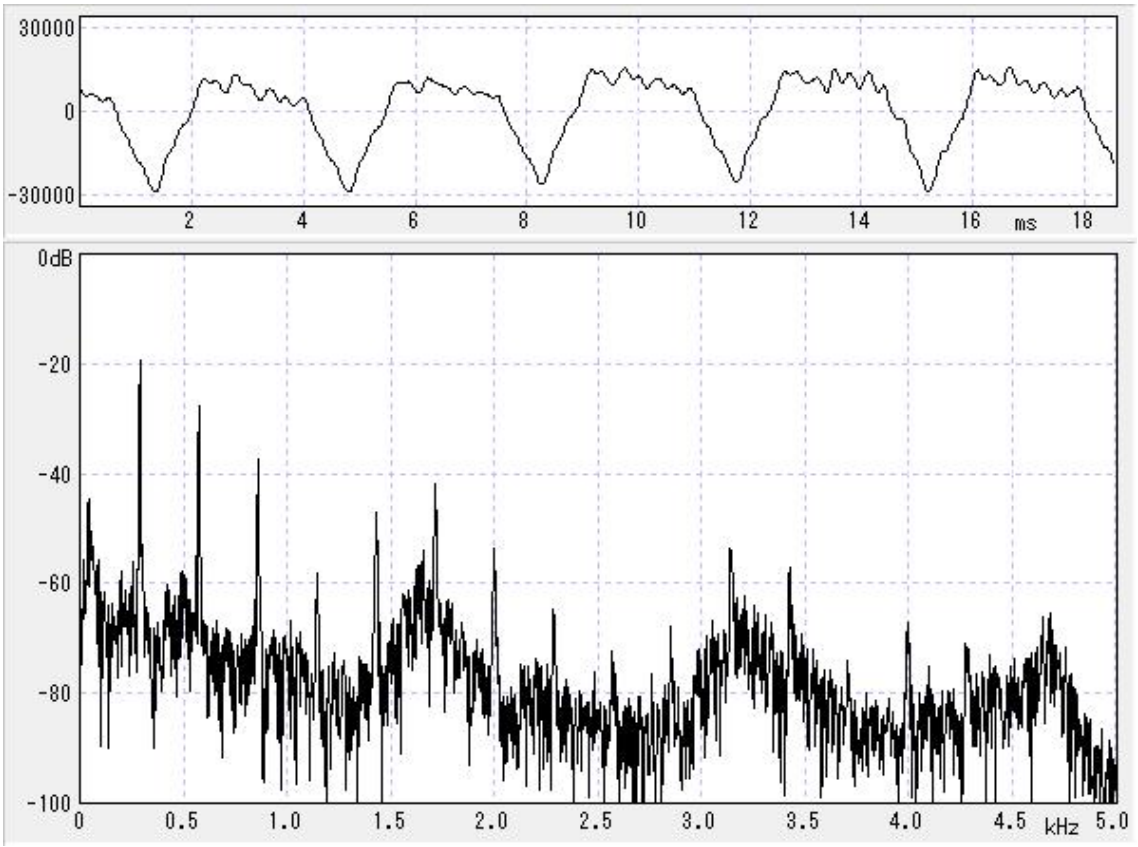

Fig2.5.3 日本語母音[u]の波形及びスペクトル

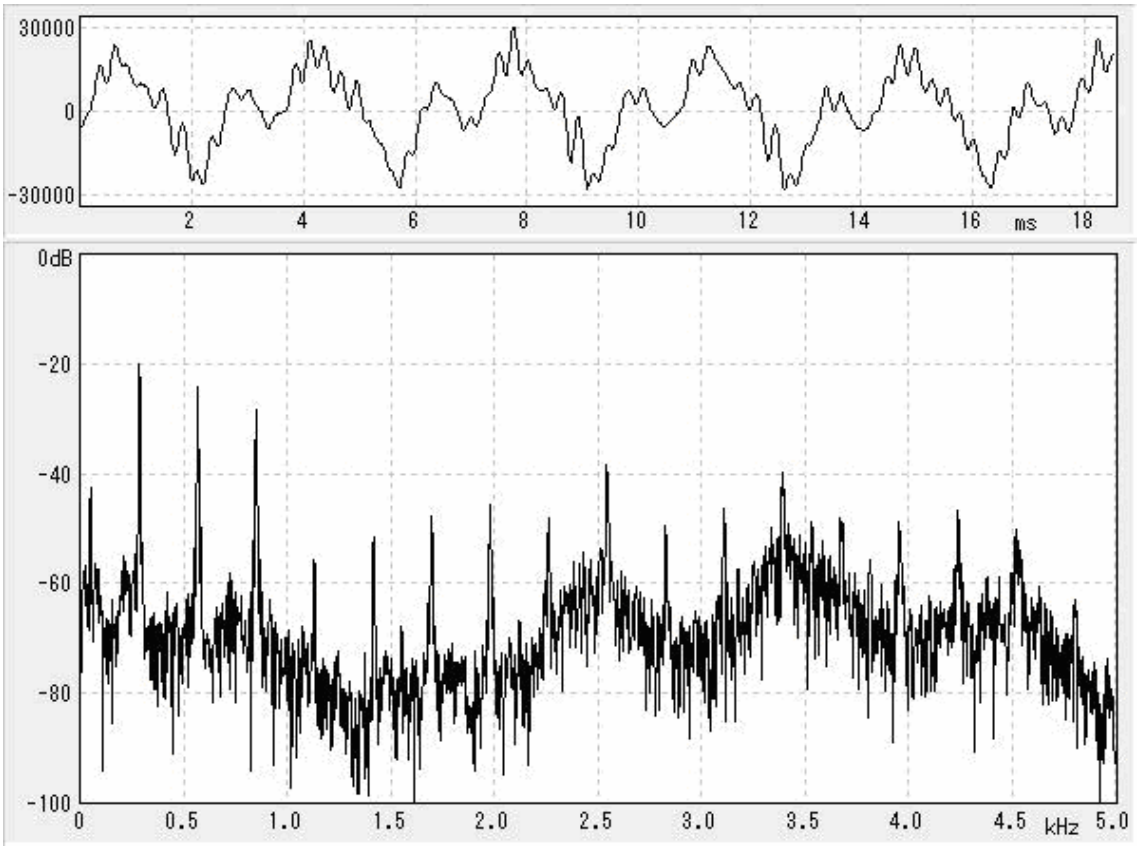

Fig2.5.4　日本語母音[e]の波形及びスペクトル

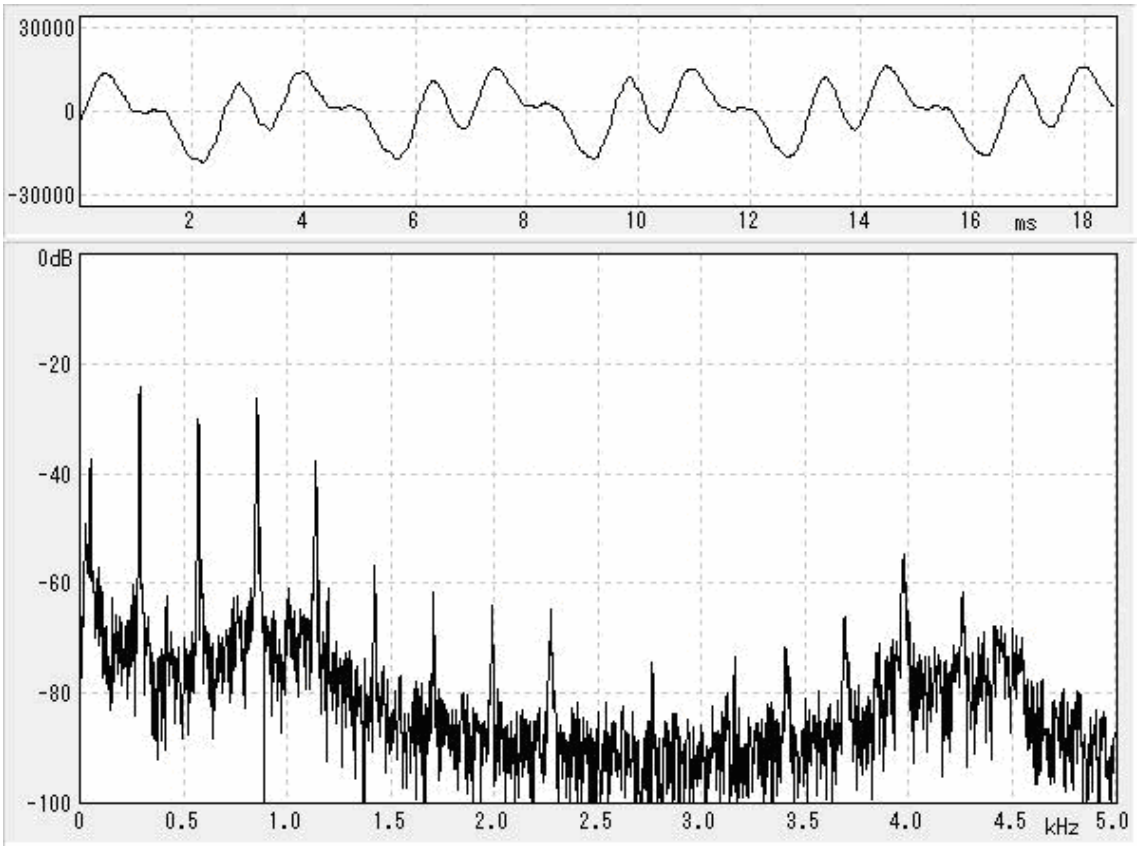

Fig2.5.5　日本語母音[o]の波形及びスペクトル

2.2.2.2 フォルマント

人間の声は振幅，帯域幅，周波数などが性別，年齢などにより一人一人違う．しかし，誰がしゃべっても言葉を聞き分けることができる．それは，声の音響的特長によるものである．ここでは，母音の音響的特性であるフォルマント(formant)について説明する．

Stumph は，楽器の音色はある一定の周波数領域の倍音が強いことによって特徴付けられるとしている．つまり，隣接する他の倍音よりも強い倍音は音色の決定に大きく貢献する．母音は声帯を振動させ，声道で共鳴させることにより特定の音色を伴って作られる．声道から発せられる音のスペクトルを測定すると，特定の周波数帯域の倍音が強調されていることがわかる．この周波数帯域のことを「フォルマント」と呼び，周波数の低い順に第 1 フォルマント，第 2 フォルマント…（以下，F_1，F_2，…）と名づけられている（参考文献[1]）．フォルマント周波数は母音ごとに異なり，これを測定することで母音の判別が可能なる．その中でも重要なのが F_1 と F_2 の相対的な位置・強度の違いである．

Fig.3.2 に日本語母音の F_1- F_2 分布図を示す．この図は横軸に F_1，縦軸に F_2 を取って，多数の人間が発声した母音を解析してまとめたものである．この図には一部重なる箇所があるものの，母音の F_1，F_2 の特徴がわかる．

これによって「音韻性」は F_1，F_2 の絶対位置よりも相対位置によって決まると考えられ，いわゆる声の男らしさ，女らしさは相対位置によって決まるということが考えられる．（参考文献[8]）

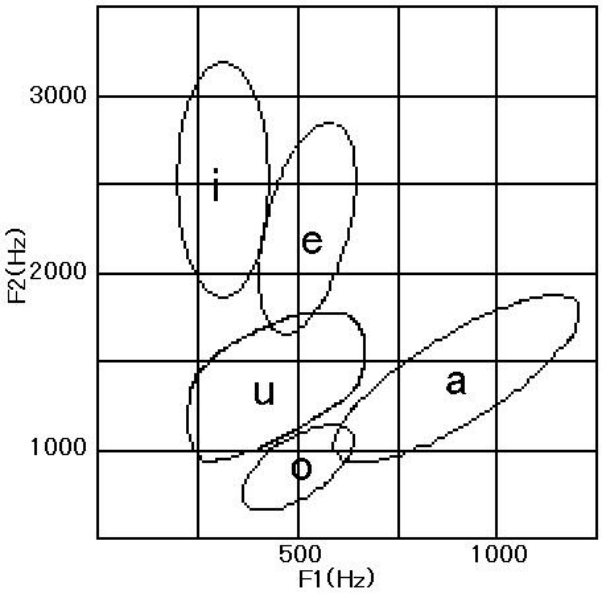

Fig2.6　日本語母音の F_1- F_2 分布図

2.2.3 子音の分類

母音はその発音が持続できる定常音であるが，子音はその音が過渡的にしか発音できない過渡音である．母音の音源は声帯で発せられるが，子音の音源はそれとは異なっている．子音の場合，肺から送られてきた空気流から音声を生成するために，声帯を振動させるのではなく，以下の二つの方法がある．

ひとつは「摩擦音」と呼ばれるものである．この音は，舌と歯，歯茎もしくは口蓋によって声道のある場所に局所的な狭めを作り，空気が通り抜ける際に乱流雑音が生成されるものである．

もうひとつは「破裂音」と呼ばれるものである．この音は，唇もしくは舌によって声道を一瞬遮断し，その後急激に声道を広げ勢いよく息を吐き出すことによって音を生成するものである．

また，この子音にはこれらの他に，「破擦音」，「鼻音」及び「半母音」と呼ばれるものがある．

「破擦音」は，「破裂音」と「摩擦音」が組み合わさったもので，破裂の動作の後すぐに摩擦音を生成するものである．

「鼻音」は，軟口蓋を下げて気道と鼻腔を接続する．そして音によって唇もしくは舌によって声道を一瞬遮断し，鼻から音を出す．

「半母音」は，声道の形状が発声の際に，ある母音の形から，別の母音の形に変化するものを言う．例えば，「や」，「ゆ」，「よ」は，「い」からそれぞれ「あ」，「う」，「お」に声道の形状が変化する．日本語子音の分類を Table2.1 に示す．

Table2.1　日本語子音の分類（参考文献[13]）

調音位置		口唇		歯，歯茎		口蓋		声門
音源		有声	無声	有声	無声	有声	無声	無声
調音方式	摩擦音		ϕ	ζ	σ	ϑ	Σ	η
	破擦音			$\delta\zeta$	$\tau\sigma$	$\delta\vartheta$	$\tau\Sigma$	
	破裂音	β	π	δ	τ	γ	κ	
	半母音	ω		ρ		φ		
	鼻音	μ		ν		N		

第3章 Singer Robot

3.1 はじめに

この章では本研究に使用しているロボット，Singer Robot について説明する．まず，ロボットの概要・動作について説明する．動作の説明では本研究の計算で使用する舌の高さの定義を説明している．その後，調音機構に求められる機能について詳しく説明する．

3.2 概要

Singer Robot の構成は、大きく分けて二つの部分に分けることができる。人間の肺・声帯の役割を持つ原音発声機構と、人間の口腔等の役割を持つ調音機構である。これは人の発声器官に由来している．それぞれの機構を Fig3.1、Fig3.2 に示す。

Singer Robot の主な特徴は２つある．１つ目は，人間の声帯の代わりにゴム膜を使用している点である．このゴム膜に空気をあてることで振動させ，音声の元になる原音を発声させている．２つ目は，声道に直方体の音響管を使用している点だ．直方体の音響管の中にある人工舌を動かすことで人間の口腔を表現し、調音をおこなう。

昨年度までの研究により、Singer Robot は[a]、[i]、[u]、[e]、[o]の５つの日本語母音を、連続的に発声させることを成功した。また，原音発声機構にはピッチ変化機構も追加し，声の高さを変化させることを可能にした．調音機構とピッチ変化機構を Fig3.3 に示す．Singer Robot のより詳しい内容については参考文献[1][3]を参照してほしい．

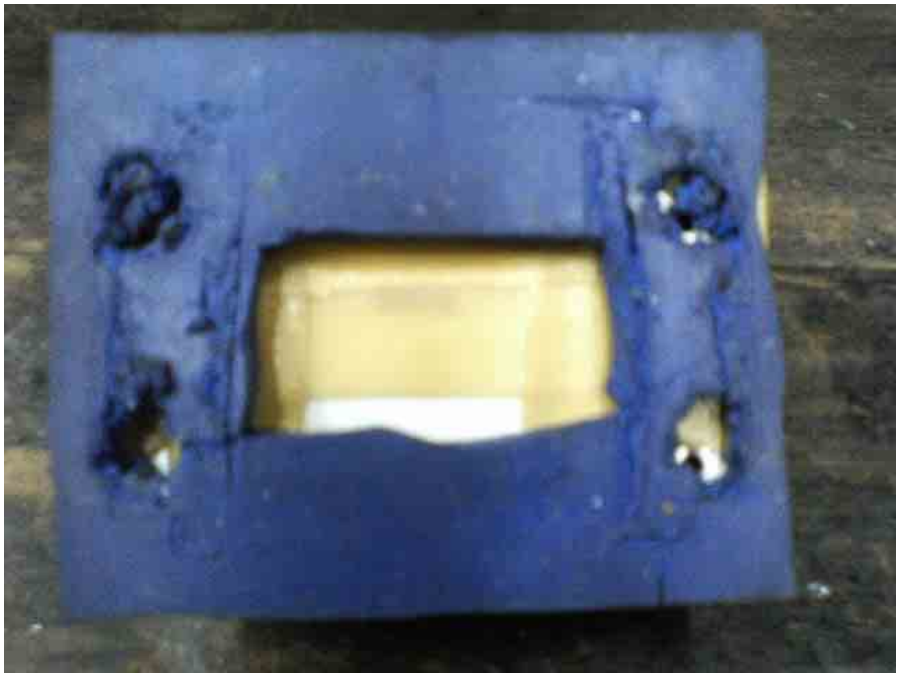

Fig3.1　原音発声機構（参考文献[1]）

Fig3.2　調音機構（参考文献[1]）

Fig.3.3　調音機構とピッチ変化機構（参考文献[3]）

3.3 動作

原音発声機構では声帯の代わりであるゴム膜を振動させ，原音を発生させている．調音機構では人工舌を動かすことで原音を調音し，母音のフォルマントを生成している．フォルマントは声道の伝達特性に影響されるが，この伝達特性は声道の断面積を変化させることで変えることができる．本研究では，直方体の音響管を声道として使用しており，その中で人工舌の高さを調節し，声道の断面積を変化させ，フォルマントの調節をおこなっている．Singer Robot の側面図を Fig.3.4 に示す．左側の模様がある長方形が原音発声装置で，右側の長方形が調音装置で，ロボットの声道にあたる．Fig.3.2，Fig.3.3 とは表示している向きが違うので注意してほしい．黒い棒が人工舌を動かすスライドで，本論文ではこの場所を駆動点と呼んでいる．駆動点の数は，人の声道の再現度や制御のしやすさを考慮し，5 箇所にしてある．5 つの駆動点は等間隔に並べるように，声帯から 40，65，90，115，140mm の距離に配置した．駆動点にはモータ，ポテンションメータが取り付けられており，上下にスライドする構造になっている．駆動点のモータを動かすことで，人工舌の高さの調節をおこなう．人工舌の高さは，破線を基準とし，破線から人工舌の上辺までの距離を x として定義している．

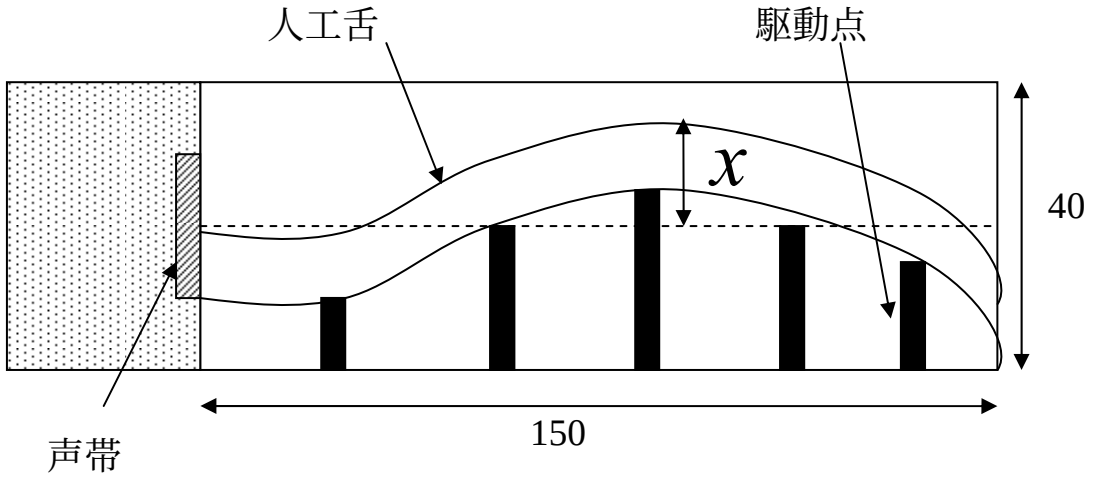

Fig.3.4　調音機構の側面図（左側：声帯側，右側：唇側）

3.4 調音機構に求められる機能

人間は声帯で原音を生成して，それを調音機構で加工して様々な音声を作り出している．原音機構で求められる機能は第 2 章で示した通りであるが，調音機構に求められる機能について詳しく説明する．音の加工は声道の断面積を様々に変化させることにより行われるが，母音を発声する場合には原音のスペクトルを変化させるフィルタの役割を果たしている. その様子を Fig3.4 に示す. 声道がある特定の断面積を持つと，特定の周波数が強調され(フォルマント)，これらの周波数が母音を識別するために重要である．まず声帯において，基本周波数の他，その整数倍の周波数を持った音である倍音が生成される．これらの音は約-12dB/oct の関係を持つ.

次に声帯で生成された音が声道を通過するが，声道は音響管としての伝達特性を持ち，共鳴を起こす．このようにして特定の周波数成分が強調され，唇から放射される.

調音機構に求められる機能は, 人間と同じように, 声道の断面積を変化させ, 伝達特性を変化させてフォルマントを作りだすフィルタの機能である．しかし, 調音器官の役割を果たす舌，唇，歯及び歯茎などから構成される人間の声道の形状は，複雑な 3 次元的形状を持っている.

ここで重要なことは，調音機構の形は必ずしも人間の調音器官の形状を再現する必要はないということである. 例えば, 九官鳥やオウムなど一部の鳥類は, 原音発声器官，調音器官が人間と著しく異なっているにもかかわらず，人間と同じような発声が可能である.

本研究では，ロボットの形状は人間の器官の形状を模倣したものではなく，ロボットに適している調音の機能を満たす形状を考えることにする．現在までの研究により, 単一音響管を用いて十分に識別可能な日本語の母音[a], [i], [u], [e]及び[o]を生成することが可能である. 単一音響管の原理の説明は省略する.

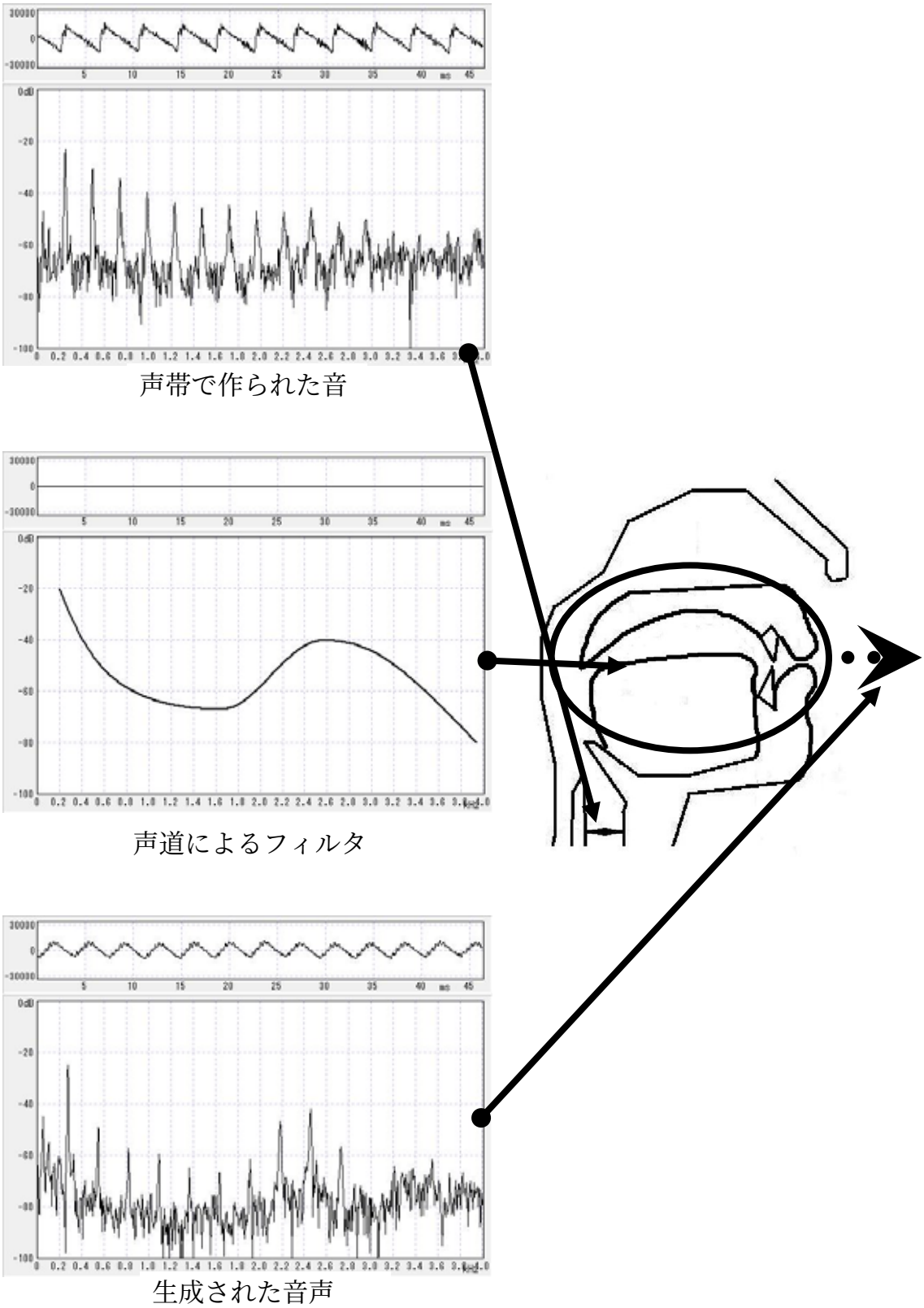

Fig3.5　調音機構によって変化するスペクトルの概念図

第4章 リアルタイムフォルマント調整システム

4.1 システムの目的

リアルタイムフォルマント調整システムの目的は，ロボットが発声した母音がより的確なフォルマントを持つように，発声した声のフォルマントを抽出し，ロボット自身にフィードバックさせることでフォルマントを再調整する，いわゆる聴覚フィードバックを行うことである．

聴覚フィードバックとは，発した音声を自分の聴覚系にフィードバックしながら，原音発声器官や調音器官の制御を行うことを言う．人は絶えず自分自身の声を耳で聞き、得られた情報にあわせて自分の声の音量や会話速度、リズム、声の調子や印象などを調整している．この聴覚フィードバックにより正常な発話が可能となっている．この聴覚フィードバックはロボットにも適応することができる．ロボットが歌うためには，母音のフォルマントを適切に調整する必要がある．聴覚フィードバックを用いることで，フォルマントの再調整が可能となり，より明瞭な発声をおこなうことができるようになる．

本研究で開発するシステムは聴覚フィードバックを再現することを目指すが，そのシステムでは「ロボットの声を一度 WAVE ファイルなどの音声ファイルとして録音してからフォルマントを抽出しフィードバックさせる」ことはおこなわない．システムの名前にある通り，「リアルタイムでフォルマントを抽出しフィードバックさせる」ことを目指している．

4.2 システム構成

リアルタイムフォルマント調整システムは大まかに「フォルマント調整プログラム」と「フォルマント抽出プログラム」に分けることができる．

「フォルマント調整プログラム」はロボットが実際に発声している母音のフォルマント F と目標となる母音のフォルマント F_0 との誤差 ΔF を最小にするように，舌の高さの変化量 Δx を算出するプログラムである．

「フォルマント抽出プログラム」はロボットの声を分析し，発声している母音のフォルマントを算出するプログラムである．ただし，ロボットの声は WAVE ファイルなどの音声ファイルに保存するのではなく，計算機内の一時メモリに保存されたデータを計算に使用する．

リアルタイム調整システムのブロック線図を示す．ブロック線図では 「フォルマント調整プログラム」と「フォルマント抽出プログラム」のほかに，ロボットとそのコントローラも表記されている．

システムの処理の流れは次のようになる．

1） ある母音を発声させるときの舌の高さ x_0 をコントローラに入力する．
2） コントローラからロボットに舌の高さに対応した電圧がかけられる．
3） ロボットが母音に対応した声道形状を再現し，母音を発声する．
4） ロボットの声をマイク等で拾い，フォルマント抽出プログラムによりフォルマント F を求める．
5） 母音をはっきり判別できるフォルマントを目標フォルマント F_0 として入力する．
6） 目標フォルマント F_0 と実際のフォルマント F との差を求め，その値 ΔF を入力値としてフォルマント調整プログラムに与える．
7） フォルマント調整プログラムから，フォルマントの誤差 ΔF をなくすように舌の高さの変化量 Δx が出力される
8） 元の舌の高さ x_0 と変化量 Δx を加算し，コントローラに送る
9） 2）にもどる

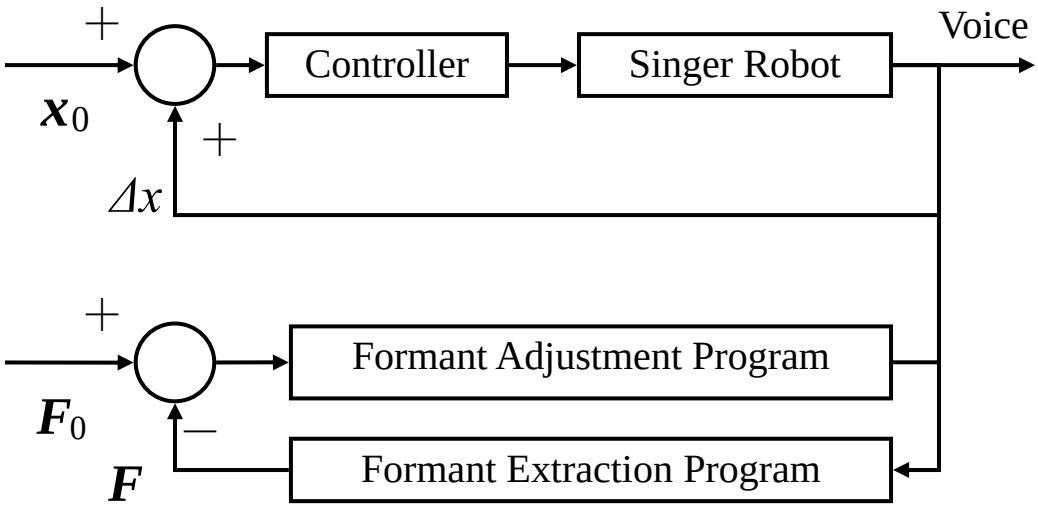

Fig.4　リアルタイムフォルマント調整システム

第5章 フォルマント調整

5.1 はじめに

リアルタイムフォルマント調整システムは「フォルマント調整プログラム」と「フォルマント抽出プログラム」に分けられる．「フォルマント調整プログラム」はロボットが発声している母音のフォルマントと目標となる母音のフォルマントとの誤差を最小にするように，舌の高さの変化量を算出するプログラムである．このプログラムを使用することで，より明瞭な母音を発声させることができる舌の高さを算出することを目指す．

この章ではまず，調整アルゴリズムの基本となる調整ベクトルの説明をする．次に，フォルマントから舌の高さを求めるという逆問題を解くために，目的関数を導出する．目的関数の最小化のために使用しているタブーサーチ法についての説明もおこなう．最後に，調整アルゴリズムのシミュレーション結果を示す．

5.2 調節ベクトル

フォルマント調整のためのアルゴリズムについて検討する．「フォルマントを調節する」ということは第 2 章の Fig.2.6 の F_1-F_2 分布図において，点を任意の位置に移動させることを意味する．本研究では，点を移動させるときのベクトルを調整ベクトル$[\Delta F_1, \Delta F_2]^T$と呼ぶことにする．調整ベクトル$[\Delta F_1, \Delta F_2]^T$は母音を発声しているときの声道の形状，つまり舌の高さによって決まる．そのため，目標となる調節ベクトル$[\Delta F_1, \Delta F_2]^T$を持つような舌の高さの変化量を求めることで，フォルマントを調整することができるのである．この調節ベクトル$[\Delta F_1, \Delta F_2]^T$は Kenneth N.Stiven（参考文献[6]）によって求められた式を利用して求める．

Steven は摂動論を用いて音響管の形状とフォルマント周波数の関係を明らかにした．摂動が小さい場合，固有振動数の変化は平均保存エネルギーの変化に比例する．つまり，フォルマントの変化量 ΔF_n は，音圧と速度が持つ運動エネルギーの変化量を足した ΔW_n に比例する．ΔW_n は次の式で与えられる．

$$\Delta W_n = -\frac{1}{4}|Pm|^2 \frac{\Delta l \, \Delta A}{\rho c^2} \cos \frac{4\pi F_n x}{c} \qquad (5\text{-}1)$$

ただし，

$F_n = \left[\frac{2n-1}{4}\right] * \frac{c}{l}$ ：フォルマント周波数

c ：音速

l ：音響管の長さ

Pm ：最大音圧

ρ ：空気の密度

Δl ：距離の変化量

ΔA ：断面積の変化量

この式から$[\Delta F_1, \Delta F_2]^T$と，声帯からの距離 l の関係を知ることができる．その関係を Fig.5.1 に示す．この図では，基準線より上に曲線がある位置で声道の断面積を小さくすると，フォルマント周波数が高くなる．基準線より下にある位置で断面積を小さくすると，フォルマント周波数が低くなる．例えば，声帯付近で声道の断面積を小さくすると，ΔF_1 と ΔF_2が両方とも増加する．また，唇付近で断面積を小さくするすると，ΔF_1 と ΔF_2が両方とも減少する．

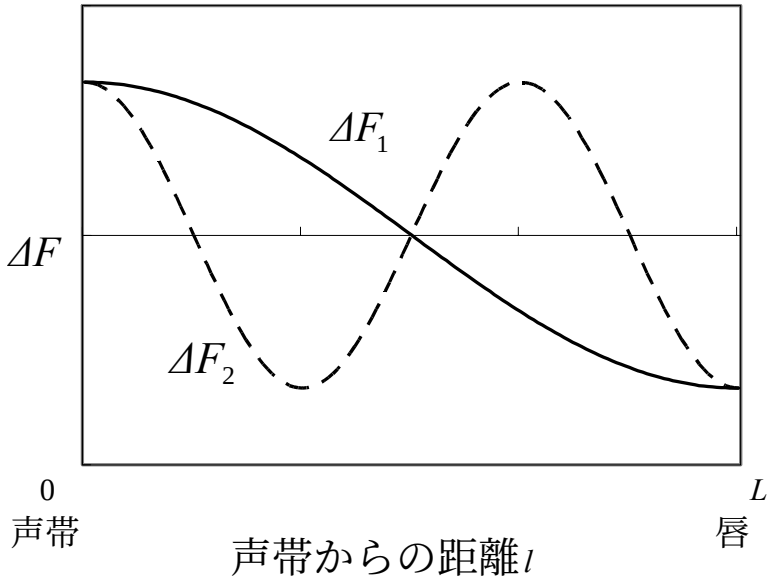

Fig.5.1　$[\Delta F_1, \Delta F_2]^T$と，声帯からの距離 l の関係

F_1-F_2 分布図での ΔF_1，ΔF_2 の関係を Fig.5.2 に示す．声帯の直前で断面積を小さくしたときのベクトルを$[\Delta F_1, \Delta F_2]^T=[1, 1]^T$とする．グラフ内の 5 つの点は調音機構の駆動点を表している．

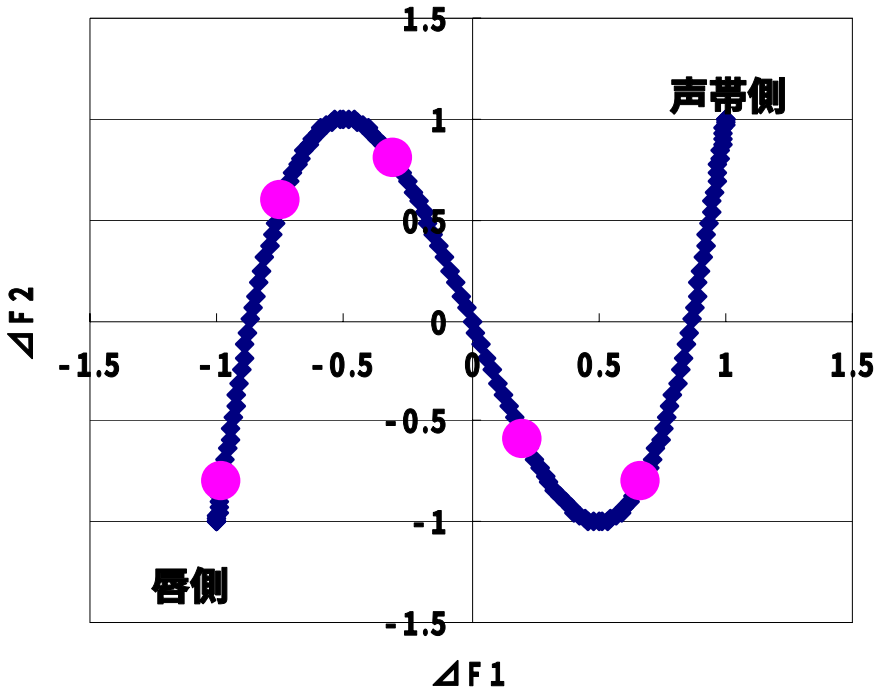

Fig.5.2　　F_1-F_2 分布図での ΔF_1，ΔF_2

5 つの駆動点での各ベクトル$[\Delta F_{1n}, \Delta F_{2n}]^T$ (n=1, 2, 3, 4, 5)に，舌の高さ x_n をかけたベクトルの和を$[\Delta F_1, \Delta F_2]^T$とする．すなわち，

$$\Delta F_i = a\sum_{n=1}^{5} \Delta F_{in} x_n \qquad (i=1, 2) \qquad (5\text{-}2)$$

a は比例定数である．Steven はフォルマントの変化量 ΔF_n と運動エネルギーの変化量 ΔW_n が比例することを示した．そのため，実際のフォルマントの変化量を求めるためには，この比例定数 a が重要になってくる．しかし，この章ではシミュレーションのみをおこなうため，a = 1 として計算し，第 7 章で改めて比例定数 a の検討をおこなう．

この式により舌の高さからフォルマント変化量を算出することができる．

5.3 目的関数

前節で示した式は舌の高さからフォルマント変化量を求める順問題である．しかし，フォルマント調整では実現したいフォルマント変化量を得るための舌の位置の変化量を求めることをおこなう．これは順問題とは変数と解が入れ替わっている逆問題を解くことになる．フォルマントの逆問題を解くのだが，この問題を直接解くことはできない. そのため, 目的関数という評価式を導出し, その評価が最小になるように計算をおこなう.

まず，想定した舌の高さ x_n で$[\Delta F_1, \Delta F_2]^T$を求め，その値と目標となる$[\Delta\hat{F}_1, \Delta\hat{F}_2]^T$との差が最小になる x_n を見つけていく. その場合の目的関数 j は次のようになる．ただし，目標となるフォルマント変化量を $\Delta\hat{F}_i$ (i=1, 2)，x から求めたフォルマント変化量の理論値を ΔF_i (i=1, 2)とすると次の式となる.

$$j = \left|\Delta\hat{F}_i - \Delta F_i\right| \qquad (i=1, 2) \qquad (5\text{-}3)$$

各母音のフォルマントは単一音響管の共鳴周波数に，声道形状から得られるフォルマント変化量を加えた値になる．フォルマントの基準となる共鳴周波数は同じ値であるため, 共鳴周波数を f_i (i=1, 2), フォルマントの目標値を $\hat{F}_i$ (i=1, 2), x から求めたフォルマントの理論値を F_i (i=1, 2)とすると,

$$\hat{F}_i = f_i + \Delta\hat{F}_i \qquad (5\text{-}4\text{-}1)$$

$$F_i = f_i + \Delta F_i \qquad (5\text{-}4\text{-}2)$$

目標関数 j は,

$$j = \left|\hat{F}_i - F_i\right| \qquad (i=1, 2) \qquad (5\text{-}5)$$

とすることができる.

5.4 拘束条件

舌の高さ x_n を求めるとき，いくつかの拘束条件を考えなければならない．本研究では拘束条件を 2 つ設定している．

(1)　駆動点の稼動範囲
声道の寸法は決まっているため，各駆動点には可動範囲がある．そのため，その範囲を超えないように舌の位置 x_n を決める必要がある．本研究ではその範囲を 15mm と設定している．

(2)　隣り合った駆動点同士の相対的な高さ
隣り合った駆動点同士の x_n の差が開きすぎると人工舌が破損する恐れがある．そのため，駆動点同士の相対的な高さも考慮する必要がある．

これらの拘束条件を満たすために，まず舌の高さの変化量 $\Delta \boldsymbol{x} = [\Delta x_1, \Delta x_2, \Delta x_3, \Delta x_4, \Delta x_5]$が大きくなり過ぎないように $\alpha \Delta \boldsymbol{x}^T \Delta \boldsymbol{x}$ 追加する．また，$\boldsymbol{x}$ が可動範囲を超えないように，$\boldsymbol{x}$ が可動範囲の限界に近づいたとき増加する値 κ を追加した．最終的な式は次の式になる．

$$J = L_1 + L_2 \tag{5-6}$$
$$L_i = \left| \hat{F}_i - F_i \right| + \alpha \Delta \boldsymbol{x}^T \Delta \boldsymbol{x} + \kappa$$

κ は次のように値を与える．可動範囲が$|x_n| < 15$ とすると，$|x_n| \leqq 14$ のときは可動範囲に余裕があるため $\kappa = 0$ とし，$14 < |x_n|$のときは$|x_n|$が可動範囲の限界に近づくにつれて κ が大きな値を持つようにするため，

$$\kappa = \left\{ \left(|x_n| - 14 \right) * 10 \right\}^2 \tag{5-7}$$

としている．

本研究で考案する調整アルゴリズムでは，この目的関数 J が最小になる舌の高さの変化量 $\boldsymbol{x}$ を求めていく．

5.5 組み合わせ最適化アルゴリズム

目的関数の最小化には「組み合わせ最適化アルゴリズム」を利用する．組合せ最適化アルゴリズムとは組合せ最適化問題の答えを導くためのアルゴリズムのことである．組合せ最適化問題は，ある集合に含まれる離散的な要素の最適な並べ方，あるいは，最適な順序を導く問題のことを言う．組み合わせ最適化アルゴリズムは，コンピュータの出現により急速に発展し，多くのアルゴリズムが考えられてきた．以下に代表的なアルゴリズムを 5 つ示す．

1） ランダムサーチ(Random Search : RS)
2） 局所的探索手法(Local Search : LS)
3） シミュレーティド・アニーリング手法(Simulated　Annealing : SA)
4） タブーサーチ手法(Taboo Search : TS)
5） 遺伝的アルゴリズム(Genetic Algorithm : GA)

目的関数の最小化で使用するアルゴリズムに求められる性能が 2 つある．1 つは「最適化するまでの処理時間がなるべく短いこと」．Singer Robot ではリアルタイムの最適調整に時間がかかると，なめらかに歌うことができないからである．そして，もう 1 つの性能は「最適解の評価が十分高いこと」．微小なフォルマントの相違が結果に大きな影響を与えるからである．以上の性能を踏まえ，本研究ではタブーサーチ手法を使用することが適切であると判断した．

5.5.1 タブーサーチ手法(Taboo Search : TS)

タブーサーチ手法は，近傍内でいくつかの解候補を選び，その中から最良な解に移動する．局所的探索手法と同じく評価の良い方向に向かっていくことになるが，違うところは評価が悪くなる解にも移動することである．しかし，このままでは同じ探索領域をループするばあいがあるため，すでに探索した解をタブーリストに残し，タブー解に戻らないように探索を進める．タブー解であっても評価が良くなる場合は，タブーを無視して移動することもできる．これにより局所的最適解に留まらずに最適解を見つけることを可能としている．

タブーサーチ手法は決定的アルゴリズムであるため，局所的最適解に落ち着いてしまうことがある．これを回避するため，探索の出発点を複数用意し，大域的最適解を求めるようにする．

プログラム上では, 初期位置として発声している母音での舌の高さ x_n を与えている. そこから, $[\Delta F_1, \Delta F_2]^T$ を生成する x_n を計算するアルゴリズムにしている.

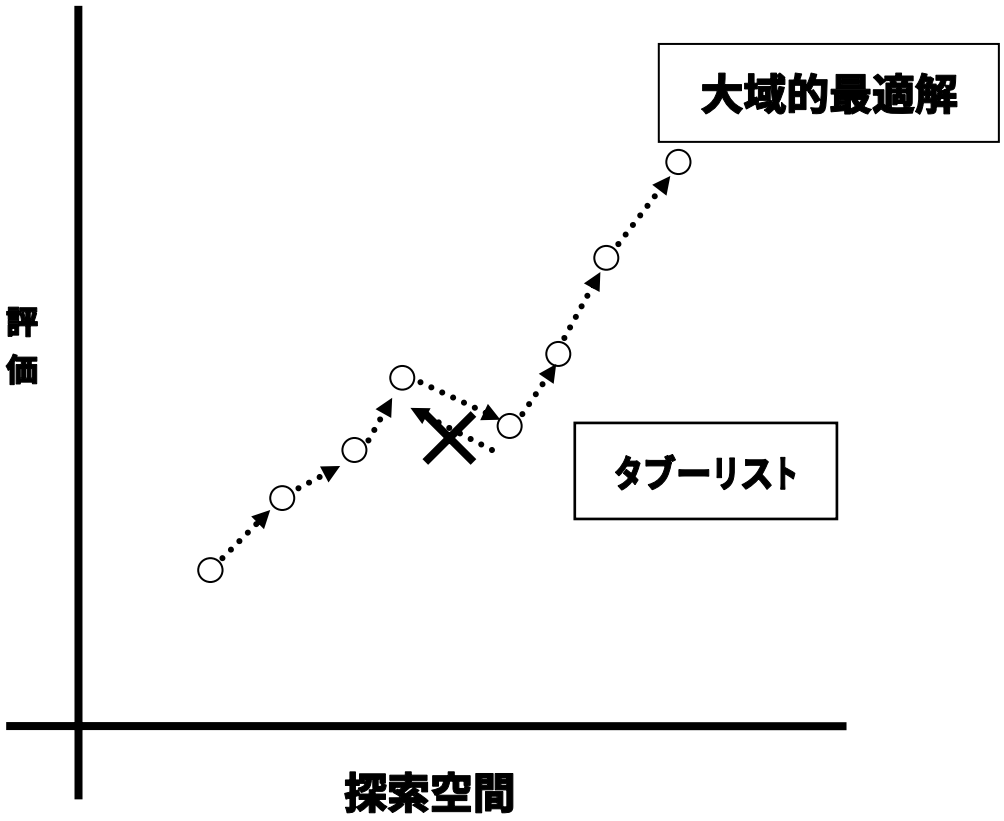

Fig.5.3 タブーサーチ手法

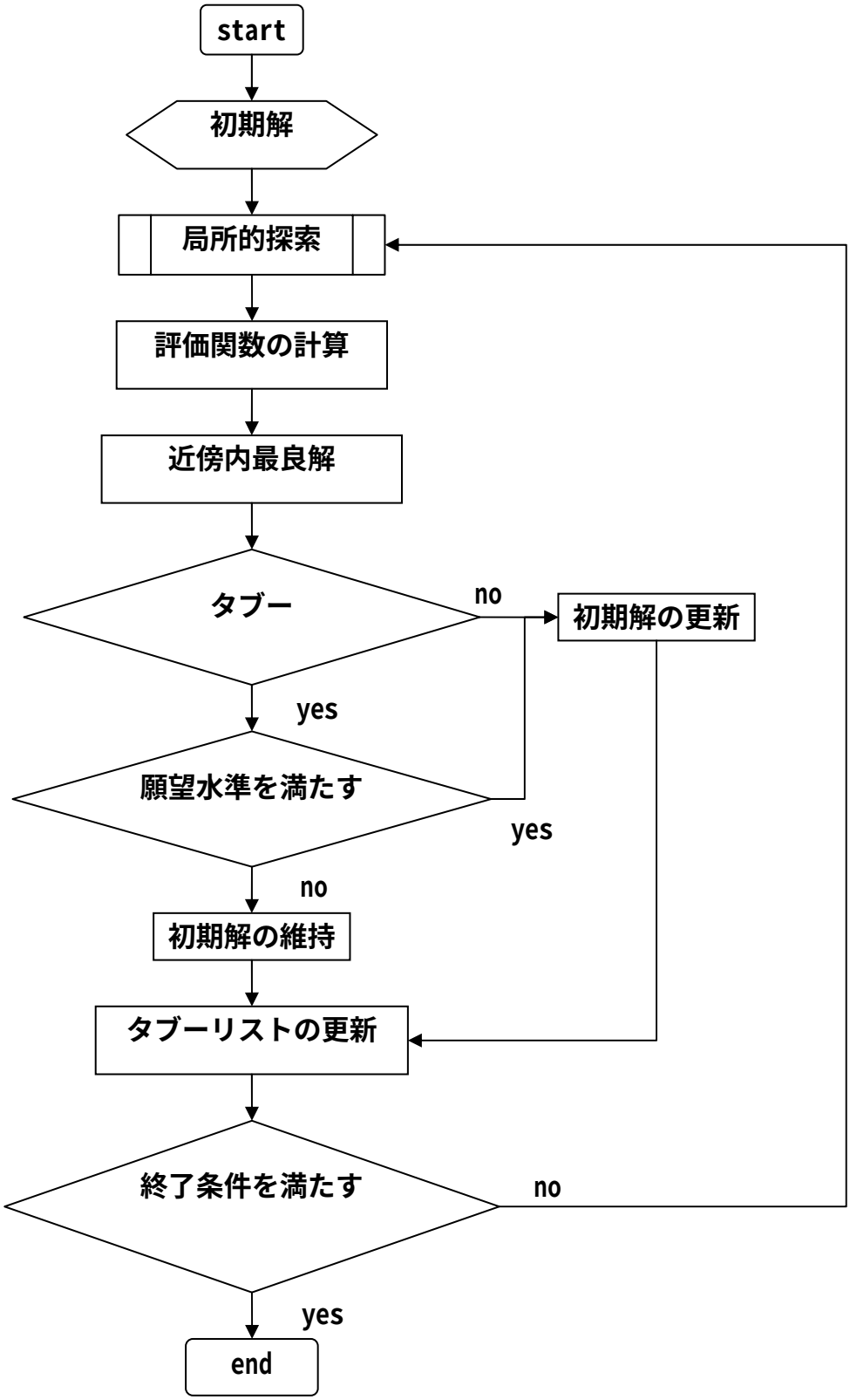

Fig.5.4　タブーサーチ手法のアルゴリズムの流れ

5.6 調整プログラム

タブーサーチ手法を用いて，フォルマント調整プログラムを書いていく．本研究ではリアルタイムフォルマント調整システムを開発することが目的である．そのため，調整する際の人工舌の初期位置は[a][i][u][e][o]を発声させるときの高さとする．その状態でのフォルマントからどのくらい変化させるかを入力する．入力では求めたい調節ベクトル$[\Delta F_1, \Delta F_2]^T$を与える．

まず，各駆動点のフォルマント変化量のベクトル$[\Delta F_{1n}, \Delta F_{2n}]^T$ (n = 1, 2, 3, 4, 5)を示す．駆動点は声帯側から 1,2,…となっている．

$[\Delta F_{11}, \Delta F_{21}]^T$ ＝[0.669130735882014, -0.809016687038484]^T
$[\Delta F_{12}, \Delta F_{22}]^T$ ＝[0.207911967850909, -0.587785939687086]^T
$[\Delta F_{13}, \Delta F_{23}]^T$ ＝[-0.309016621414425, 0.809016302867593]^T
$[\Delta F_{14}, \Delta F_{24}]^T$ ＝[-0.743144490185651, 0.587786468451883]^T
$[\Delta F_{15}, \Delta F_{25}]^T$ ＝[-0.978147473903929, -0.809015918696358]^T

このベクトルを使い，人工舌の高さからフォルマントを計算する．

タブーサーチ法のパラメータは次のとおりである．

計算回数　：　100 回
解候補　：　30 個
タブーリスト　：　5，15 個（探索状況によって変動する）

製作したプログラムでフォルマント調整をシミュレーションする．初期位置が母音[a]の状態から，$[\Delta F_1, \Delta F_2]^T$がプラスになるように入力する．$F_1$-$F_2$ 分布図で考えると，フォルマントの位置が右上に移動すること想定している．（Fig.5.4）

プログラムの実行結果を Fig.5.5.1 と Fig.5.5.2 に示す．Fig.5.5.1 は調整前を，Fig.5.5.2 は調整後を表している．舌の形状を見ると，調整後は大まかな形状は変わっていないが，5 番の駆動点（一番唇に近い駆動点）が舌に下がっていることがわかる．母音の舌の形状を大きく変えることなく，フォルマントを調整することができたといえる．

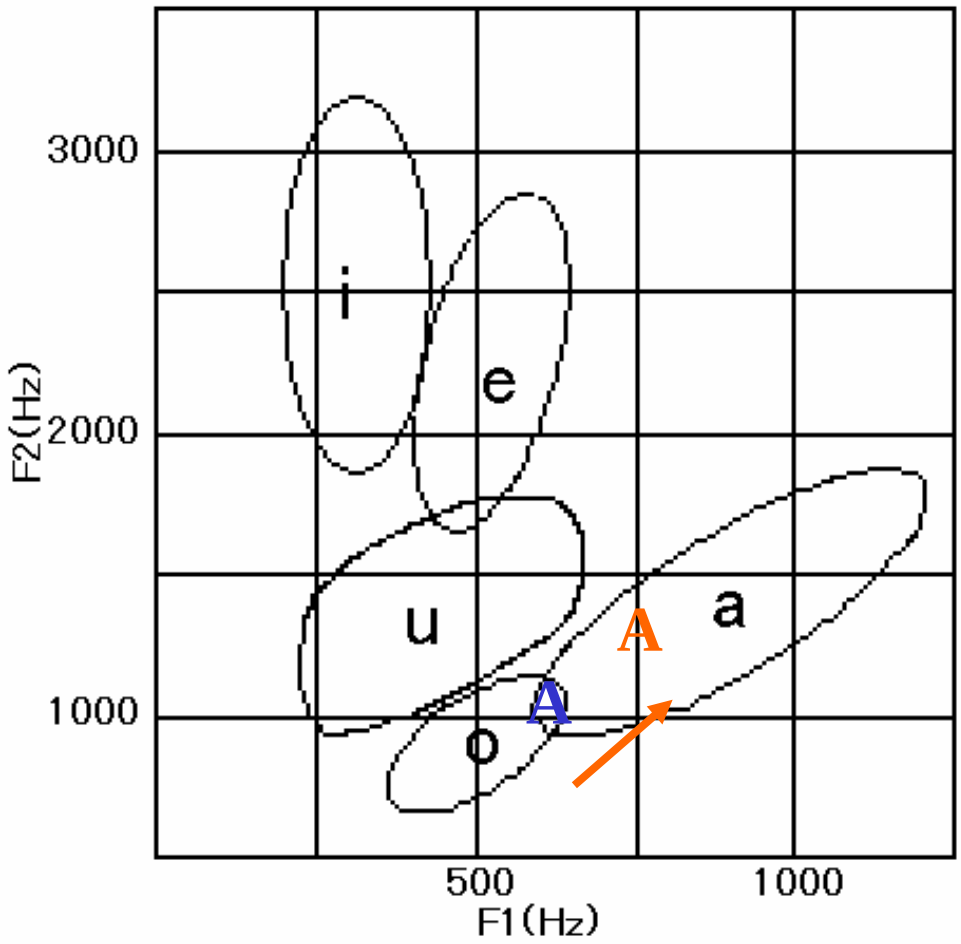

Fig.5.4　　F_1-F_2 分布図での移動方向

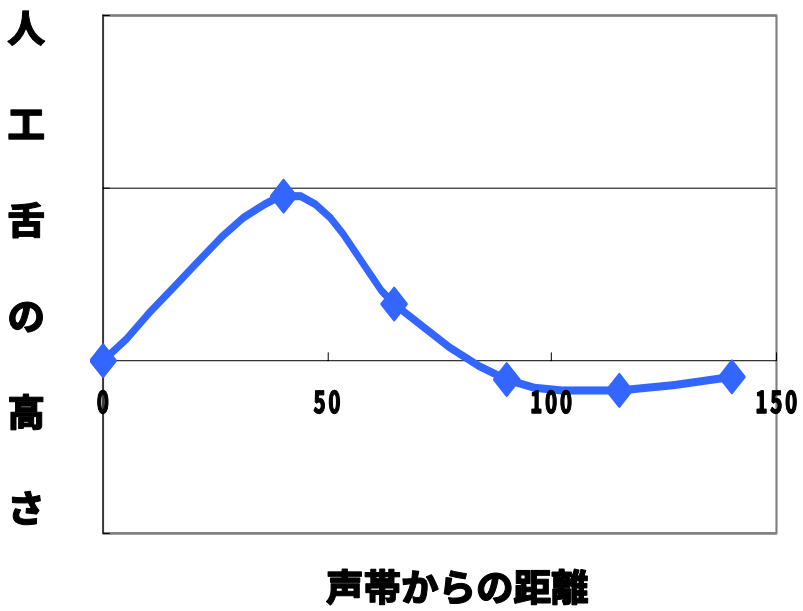

Fig.5.5.1　母音[a]の調整前の舌の形状

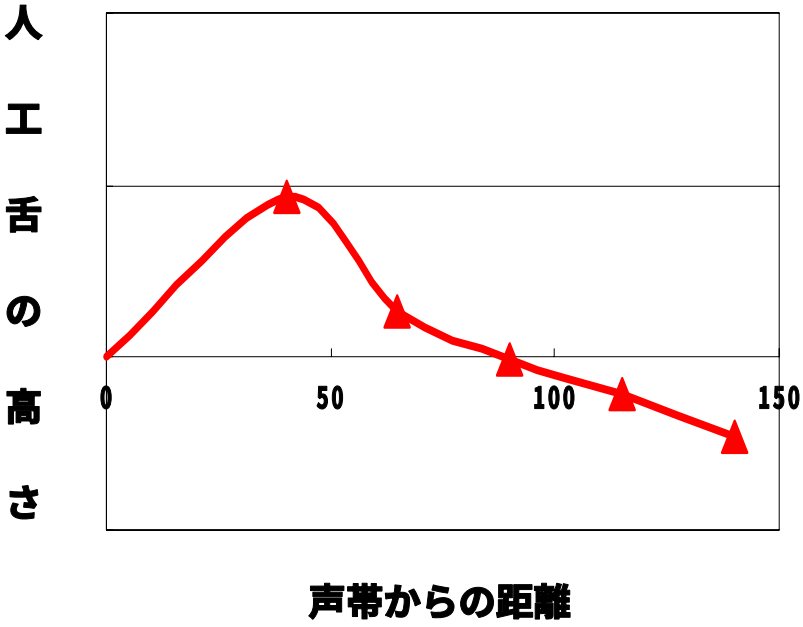

Fig.5.5.2　母音[a]の調整後の舌の形状

5.7 おわりに

この章ではリアルタイムフォルマント調整システムの「フォルマント調整プログラム」について述べた．調整ベクトル$[\Delta F_1, \Delta F_2]^T$と言う考え方から，フォルマントの変化量から舌の高さの変化量を算出するアルゴリズムを開発することに成功した．

目的関数の最小化ではタブーサーチ法を用いたが，計算精度の向上や処理時間の短縮のために，パラメータの設定はさらに検討をおこなっていく．

作成した調整プログラムは, 現在の Singer Robot に採用されている直方体の声道に対応させてある．そのため，声道の形状を円柱状の声道に変更する場合は式(5-2)を変更する必要がある．

第6章 フォルマント抽出

6.1 はじめに

聴覚フィードバックをおこなうために，ロボットが発声した音声からフォルマントを抽出するプログラムが必要である．「フォルマント抽出プログラム」はマイクから音声を取得し，発声している母音のフォルマントを算出するプログラムである．ただし，本研究では音声は WAVE ファイルなどの音声ファイルに保存するのではなく，計算機内の一時メモリに保存されたデータを計算に使用することを目指す．

この章では，音声からフォルマントを求める手順について述べていく．フォルマント抽出の手段として線形予測法を紹介し，次に音声処理のときの前処理を説明する．その後，線形予測法の検証を MATALAB によりおこない，最終的に C 言語で Windows アプリケーションを作成する．

6.2 フォルマント抽出

フォルマント抽出には，フーリエ分析と線形予測法が挙げられる．線形予測法とは「任意の標本が直前の標本から部分的に予測可能である」という事実に基づき，全極型モデルの拘束条件で音声分析する方法である．フーリエ分析によるスペクトルは基本波に対する高調波を表しており，線形予測法によるスペクトルはフォルマント周波数と振幅を表している．フーリエ分析はさまざまな周波数成分を表すことができるが，フォルマントについては熟練者でなければ正確に推定するのが困難である．それに対し，線形予測法は容易にフォルマントを推定することができる．また，計算量もフーリエ分析より線形予想法のほうが少ない. Fig.6.1 は母音[i]をフーリエ分析と線形予測法で分析した図である．黒い細線はフーリエ分析を，赤い太線は線形予測法を示している．

本研究ではフォルマントのみを分析することを目的としているため，線形予測法を利用することにした．

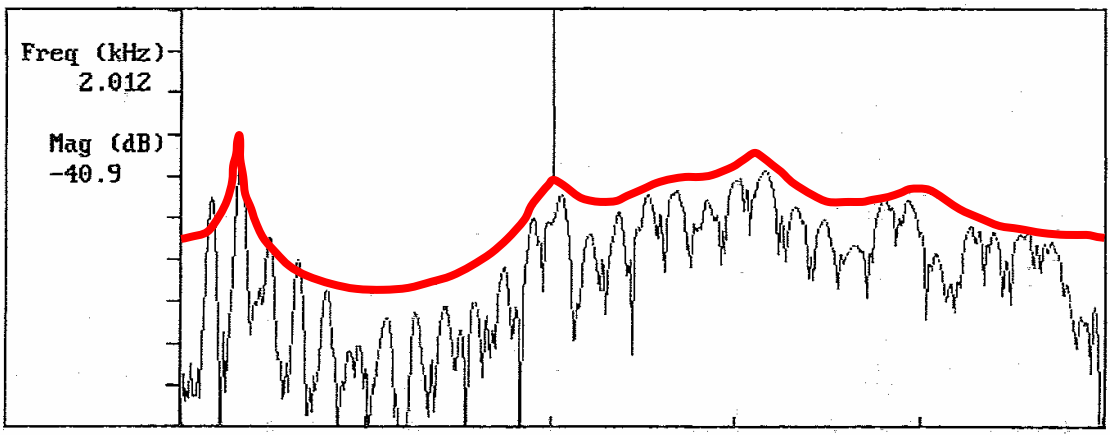

Fig.6.1　フーリエ分析と線形予測法（参考文献[9]）

6.2.1 線形予測法（LPC）

本研究で，フォルマント抽出にて使用する線形予測法の計算方法について説明する．

線形予測法は，現在の出力サンプル x_t がそれ以前の N 個のサンプル値 x_{t-n}，n = 1, 2, 3,…, N の線形結合によって予測されるものとするモデルを考える．いまモデルの出力信号波形を x_t とし未知入力を u_t とするとき

$$x_t = -\sum_{n=1}^{N} \alpha_n x_{t-n} + G u_t \tag{6-1}$$

と定式化される．x_t，u_t の z 変換をそれぞれ $X(z)$，$U(z)$とすると，上の式の z 変換より伝達関数は

$$H(z) = \frac{X(z)}{U(z)} = \frac{G}{1 + \sum_{n=1}^{N} \alpha_n z^{-n}} \tag{6-2}$$

この伝達関数を全極型モデルと呼ぶ.

一方，母音の生成過程は，ラプラス演算子 s を用いて，声道の伝達特性を $V(s)$, 音源特性を $S(s)$，唇からの放射特性を $R(s)$とすると

$$T(s) = R(s) \cdot V(s) \cdot S(s) \tag{6-3}$$

の形で表すことができる．このうち，$R(s)$，$S(s)$は近似的にピークを持たない傾斜スペクトルによって表現できるので，伝達関数 $T(s)$のスペクトル包絡はほぼ $V(s)$によって特徴づけられる．その $V(s)$の z 変換形は全極型モデルと同じ形になる．そして，この伝達関数の極，分母＝0 となる解がフォルマントに対応している．その解を z_k とすると

$$f_n = \frac{1}{2\pi T} \mathrm{Im}\left[\log z_n\right] \tag{6-4}$$

がフォルマントを表している．次数 N は経験的な値が採用されており，成人男声の場合，標本化周波数 8kHz で 12，10kHz で 14，16kHz で 20 程度であり，女声では 1〜2 割程度少ない.

次に，伝達関数に含まれる予測係数 α_n を求めるため，全極型モデルの解法についても調査した．全極型モデルでは

$$\hat{x}_t = -\sum_{n=1}^{N} \alpha_n x_{t-n} \tag{6-5}$$

$$e_t = G u_t \tag{6-6}$$

とおくと式(6-2)は

$$e_t = x_t - \hat{x}_t \tag{6-7}$$

と書け，e_t を予測誤差と見ることができる．e_t の平均 2 乗差を最小にする評価基準によって，α_n を決定する．平均予測誤差は

$$E_N = E(e_t^{\,2}) = E[(x_t + \sum_{n=1}^{N} \alpha_n x_{t-n})^2] \tag{6-8}$$

ここで $E(\cdot)$は期待値をとることを表している．この最小化は

$$\frac{\partial E_N}{\partial \alpha_i} = 0 \qquad (i = 1, 2, 3, \ldots, N) \tag{6-9}$$

によって得られ，次の N 個の連立方程式が得られる．

$$\sum_{n=1}^{N} \alpha_n E(x_{t-n} x_{t-i}) = -E(x_t x_{t-i}) \qquad (i = 1, 2, 3, \ldots, N) \tag{6-10}$$

これを

$$E(x_{t-n} x_{t-i}) = r_{n-i} = r_{i-n} \tag{6-11}$$

と置くと，次のように書くことができる．

$$\begin{bmatrix} r_0 & r_1 & \cdots & r_{n-1} \\ r_1 & r_0 & \cdots & \vdots \\ \vdots & \vdots & \ddots & r_1 \\ r_{n-1} & \cdots & r_1 & r_0 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} = - \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} \tag{6-12}$$

この方程式をユール・ウォーカの方程式と呼ぶ．この方程式をダービンの解法を用いて解き，α_n を求める．

6.2.2 線形予測法の前処理

線形予測法によってフォルマントを求める際，マイクなどからのサンプリングデータをそのまま計算してはいけない．なぜなら，音声の伝達特性やデータの連続性を考慮する必要があるからだ．そのため，サンプリングデータにいくつかの処理をおこなってから計算する．その前処理にはプリエンファシス，フレームのブロック化，窓掛けがある．

6.2.2.1 プリエンファシス

プリエンファシスとは，伝送の過程で周波数が高いほどレベルが低下しやすいことから発生する信号の品質劣化を補うための技術である．人間の音声の長時間スペクトルは，-6dB/oct で広域が減少している．そのため，ディジタル化された音声信号 $s(n)$を,低次のディタルシステム（典型的には１次の FIR フィルタ）に通して，スペクトル的に平坦化し,後の信号処理での有限演算語長の影響を少なくする．プリエンファシスの実現方法として,元のサンプル値系列に演算を施す方法を採用する．元のサンプル値を $s(n)$，演算後のサンプル値を $\tilde{s}(n)$ とすると次のようになる．

$$\tilde{s}(n) = s(n) - \tilde{a}s(n-1) \tag{6-13}$$

$\tilde{a}$ は係数で，最も普通に用いられる値は 0.95 近辺である．$\tilde{a}$ は時間とともに変化させる必要があるが，本研究では$\tilde{a}$ を 0.95 とした．

6.2.2.2 フレームのブロック化

プリエンファシスされた音声信号$\tilde{s}(n)$ を S サンプルのフレームにブロック化

する．また，隣のフレームとは M サンプル分離れるようにする．$M \leqq S$ ならば隣り合うフレームは重なり合い，LPC スペクトル推定はフレームごとに相関を持つ．もし $M > S$ ならば隣接フレーム間に重なりはなく，音声信号の幾分かはまったく失われてしまうことになり，分析結果の隣接フレームの LPC スペクトル推定の間の相関には，M の増加に伴って増大するような雑音成分を含むことになる．l 番目の音声フレームを $x_l(n)$とし，音声信号全体には L フレームあるとすれば，

$$x_l(n) = \tilde{s}(Ml + n) \qquad (6\text{-}14)$$
$$(n = 0, 1, 2,\ldots, S\text{-}1,\ \ l = 0, 1, 2,\ldots L\text{-}1)$$

となる．今回，リアルタイムでのフォルマントの抽出を行なうわけだが，常に抽出を行なうのではなく，ある一定間隔で抽出を行なうため，隣接フレームの LPC スペクトル推定の間の相関は考えない．

6.2.2.3 窓掛け

窓掛けとは，フレームに窓関数をかけて，フレームごとのはじめと終わりの部分での不連続性を最小にすることである．音声分析では得られたサンプルの一部を切り出して解析を行なう．しかし,フーリエ変換や線形予測法はサンプル値とその前後の値の関係が影響するため，切り出したフレームの両端の不連続性が解析結果に影響を与える．そのため，信号の各フレームの最初と最後で 0 になるように窓関数によって傾きをつけるのである．サンプル数（標本化点数）を S，窓関数を $w(n)$，$0 \leqq n \leqq S\text{-}1$ とすると信号に窓をかけた結果は

$$\tilde{x}_l(n) = x_l(n)w(n) \quad (0 \leqq n \leqq S\text{-}1) \qquad (6\text{-}15)$$

である．

窓関数には矩形窓やガウス窓，ハニング窓，ハミング窓などがあるが，ここでは LPC の自己相関法の典型的な窓関数であるハミング窓を使用する．使用したハミング窓関数は次式である．

$$w(n) = 0.54 - 0.45\cos\left(\frac{2\pi n}{N-1}\right) \qquad (6\text{-}16)$$

6.3 MATLAB による検証

線形予測法を使いフォルマントを求めることができるか検証する．音声データの処理には MATLAB を用いた．MATLAB には音声処理の関数が多数用意されてあり，線形予測法をおこなう LPC 関数も存在する．この LPC 関数を使い，検証用のプログラムを作成した．今回の検証では，まず WAVE ファイルからサンプリングデータを取得しフォルマントの抽出をおこなう．その次に，マイクからの入力からサンプリングデータを取得し，フォルマントを抽出する．

まず，WAVE ファイルからのフォルマント抽出をおこなう．MATLAB の wavread 関数（音声ファイルからサウンドデータを取得する関数）を使い, WAVE ファイルからサンプリングデータを取得する．得られたサンプリングデータ x_t にプリエンファシスと窓掛けの前処理を行なう．そのデータから MATLAB の LPC 関数を使い，予測係数 α_n を求める．そして，全極型モデルの伝達関数の極を求める．プログラムの各パラメータは標本化時間 τ ＝ 30ms, 標本化周波数 T ＝ 11025Hz, 標本化点数 s ＝ 330, 次数 N ＝ 10 とした. WAVE ファイルは Singer Robot が発声した母音[a]，[i]，[u]，[e]，[o]が録音されたファイルを使用した．フォルマントの抽出結果を Fig.6.2 に示す．Fig.6.2 において点が集中している部分が母音である. Fig.2.6 の F_1-F_2 分布図と比較すると，[o]以外は WAVE ファイルから抽出したフォルマントが Fig.2.6 の各分布範囲に近いことがわかる．[o]については Singer Robot が最適なフォルマントを出せていないため，大きく外れてしまっている. これは Singer Robot の調整がうまくできていないことが原因である.

次に，マイクからの入力データからフォルマントを抽出する．ここでは人間の声からフォルマントを抽出する．MATLAB では wavrecord 関数（マイクなどのオーディオ入力デバイスからデータを取得する関数）を使い計算をおこなっている．線形予測法のパラメータは上記と同じである．フォルマントの抽出結果を Fig.6.3 に示す．Fig.6.3 では母音 1 つに対して 20 個プロットをしている．Fig.2.6 と比較すると，すべての母音が範囲に入っていることが分かる.

この結果から，線形予測法でのフォルマント抽出が可能であることがわかる.

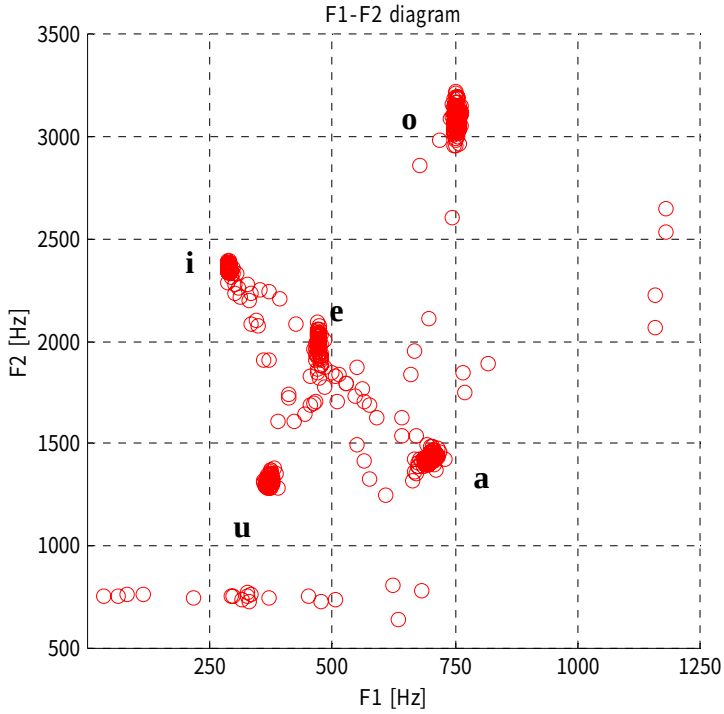

Fig.6.2　Singer Robot の声からのフォルマント抽出結果

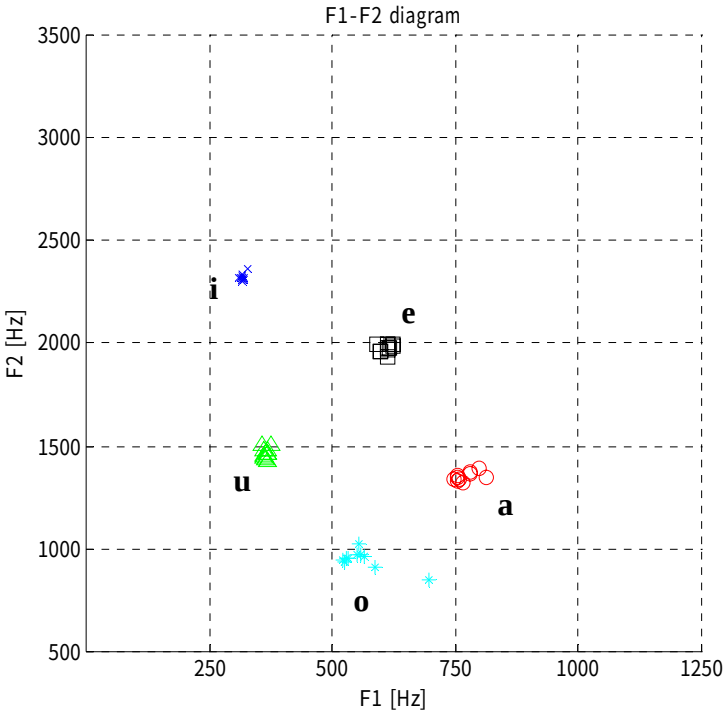

Fig.6.3　人の声からのフォルマント抽出結果

6.4 Windows アプリケーション

6.4.1 C言語への書き換え

MATLAB を用いて音声からフォルマントを求めることができた．次に，このプログラムをC言語に書き直すことをおこなう．

MATLAB のプログラムをC言語に移植する理由は，2 つある．１つは，第 5 章で作成した調整プログラムと組み合わせたときに，データのやり取りをやりやすくするためである．MATLAB からCプログラムを呼び出すことや，逆にCプログラムから MATLAB を呼び出すことは可能である．しかし，mex ファイルの作成や MATLAB エンジンのインクルードなどの準備を必要とし，データのやり取りが複雑になる．そのため，すべてのプログラムをC言語のみで作成したほうが計算しやすいと考えた．

2 つ目は，音声を録音することと，音声データを処理することを同時におこなうことが MATLAB ではできないためである．MATLAB では音声データを取得するときは wavrecord 関数を使用する．この関数でマイクから音声データを取得したあとに，前処理や LPC によりフォルマントを求めている．フォルマントを求めているときは wavrecord 関数が呼ばれていないため，録音が計算をするごとに途切れてしまうことになる．一方，Visual C++では音声の録音には低レベル API である waveInOpen，waveInPrepareHeader，waveInAddBuffer，waveInStart を使用する．その手順は，waveInOpen でマイクなどのオーディオ入力デバイスを開く．次に waveInPrepareHeader で音声データを格納する入力バッファを準備し，waveInAddBuffer で入力デバイスに準備したバッファを渡す．最後に waveInStart で音声の入力を開始する．MATLAB が 1 つの関数で済むのに対して行なう手順はかなり多く，変数の宣言が複雑である．しかし，入力デバイスに複数のバッファを渡すことで，音声データの録音と計算を同時に行なうことができる．音声の入力はデバイスに渡されたバッファに対し FIFO(First In, First Out)順で行なわれる．つまり，先に渡されたバッファに入力が行なわれ，そのバッファが先に返されるのである．よって，複数のバッファが入力デバイスに渡されたときは，データの入力中のバッファが一杯になると，そのバッファはデバイスからアプリケーションに返され，キューに順番待ちをしているバッファに入力が切り替わり，録音を開始することになる．アプリケーションに返されたバッファは計算に使用することができ，その間も録音は続けられる．

この二つの点から，今回はC言語のみでプログラムを作成することにした．

6.4.2 録音プログラム

Visual C++を用いてプログラミングをおこなう．音声の録音再生をおこなうため，Windows のマルチメディア API を使用する．音声は WAVEFORMATEX 構造体の設定に従ってサンプリングがおこなわれる．

6.4.2.1 WAVEFORMATEX 構造体

WAVEFORMATEX 構造体とは音声データをアプリケーションが取り扱う時に，モノラル・ステレオの区別や，サンプリング周波数など録音条件を決定するものである．演奏システムのアプリケーションを製作する際にも，実際に条件を決めてプログラミングをしなければならない．ここで，製作したアプリケーションの条件を含め，WAVEFORMATEX 構造体について説明する．

以下に WAVEFORMATEX 構造体と各部位の説明を，Table 6.1 に本研究で決定した条件を示す．本研究では，フォルマントを計算するだけであるため，nChannels はモノラルにした．nSamplesPerSec は線形予測法の計算の関係上，11025Hz を採用する．

WAVEFORMATEX 構造体

```
typedef struct {
        WORD    wFormatTag;          ←――― (1)
        WORD    nChannels;           ←――― (2)
        DWORD nSamplesPerSec;        ←――― (3)
        DWORD nAvgBytesPerSec;       ←――― (4)
        WORD    nBlockAlign;         ←――― (5)
        WORD    wBitsPerSample;      ←――― (6)
        WORD    cbSize;              ←――― (7)
} WAVEFORMATEX;
```

(1)　WORD　wFormatTag

WAVE データの表現形式で, 最も基本的なものが PCM という形式である. PCM は録音した波形を何も加工せずに，そのまま記録する．これによりデータサイズが大きくなある欠点を持っているが，演算処理にそのまま使用することができる．

(2)　WORD　nChannels

録音においてチャンネルをモノラル（1ch）かステレオ（2ch）かを決定する．モノラルの場合は 1，ステレオの場合は 2 である．

(3)　DWORD nSamplesPerSec

サンプリング周波数であり，単位はヘルツ（Hz）である．サンプリング周波数の半分の大きさが録音可能周波数となる．また，Windows では標準として，44100Hz，22050Hz，11025Hz の値が使用されている．

(4)　DWORD nAvgBytesPerSec

1 秒当たりのデータ転送量であり，単位は Bytes である．1 秒の音声を録音・再生するためには以下の式で求まるバッファ（Bytes / sec）が必要になる．

$$nAvgBytesPerSec = \frac{nSamplesPerSec \cdot NChannels \cdot wBitPerSample}{8}$$

(5)　WORD　nBlockAlign

音声データの 1 サンプル点を再現する際に，データが途中で分割されないようにする最小のサイズであり，単位は Bytes である．例えば wBitsPerSample が 6Bit でモノラルの場合, nBlockAlign は 3Bytes になる. それにより, 24Bit（3Bytes）のブロックに, 4 サンプル点（6Bit×4）が収納される．また PCM データの場合, wBitsPerSample が 8 の整数倍になるため，nBlockAlign は 1 サンプル点を再現するデータ量と等しくなる．

(6)　WORD　wBitsPerSample

1 チャンネル当たり 1 サンプル点を再現するデータ量であり，単位は Bit である．PCM の場合，8Bit かもしくは 16Bit を使用する．また，8Bit で WAVE 分解能は 256，16Bit で 65536 であり，ダイナミックレンジはそれぞれ，約 48dB と約 96dB となる．

(7)　WORD　cbSize

拡張型のフォーマットのエリアサイズであり，単位は Bytes である．WAVEFORMATEX に追加情報が必要な場合のみの使用となり, PCM では 0 とする．

Table 6.1　WAVEFORMATEX 構造体

wFormatTag	WAVE_FORMAT_PCM
nChannels	1 (MONO)
nSamplesPerSec	11025
wBitsPerSample	16
nBlockAlign	nChannels * wBitsPerSample / 8
nAvgBytesPerSec	nSamplesPerSec * nBlockAlign
cbSize	0

6.4.3 動作テスト

作成した Windows アプリケーションの動作テストを行なう．プログラムの各パラメータは MATLAB のプログラムと同じように，標本化時間τ ＝ 30ms，標本化周波数 T ＝ 11025Hz，標本化点数 s ＝ 330，次数 N ＝ 10 とした．

作成したプログラムの実行画面を Fig.6.4 に示す．この画面は日本語母音[a]のフォルマントを抽出している状態である．実行画面には第 1 フォルマントから第 4 フォルマントまでを表示している．左側には本プログラムに送られてくるメッセージを表示している．

フォルマント抽出結果では，第 1 フォルマント(F1)が 764Hz，第 2 フォルマント(F2)が 1305Hz となった．Fig2.6 の F_1-F_2 分布図で確認すると，[a]の範囲に入っていることが分かる．また，KTH(スウェーデン王立工科大学)が配布している音響分析ソフト「WaveSurfer」によるフォルマント抽出結果ともほぼ一致した．

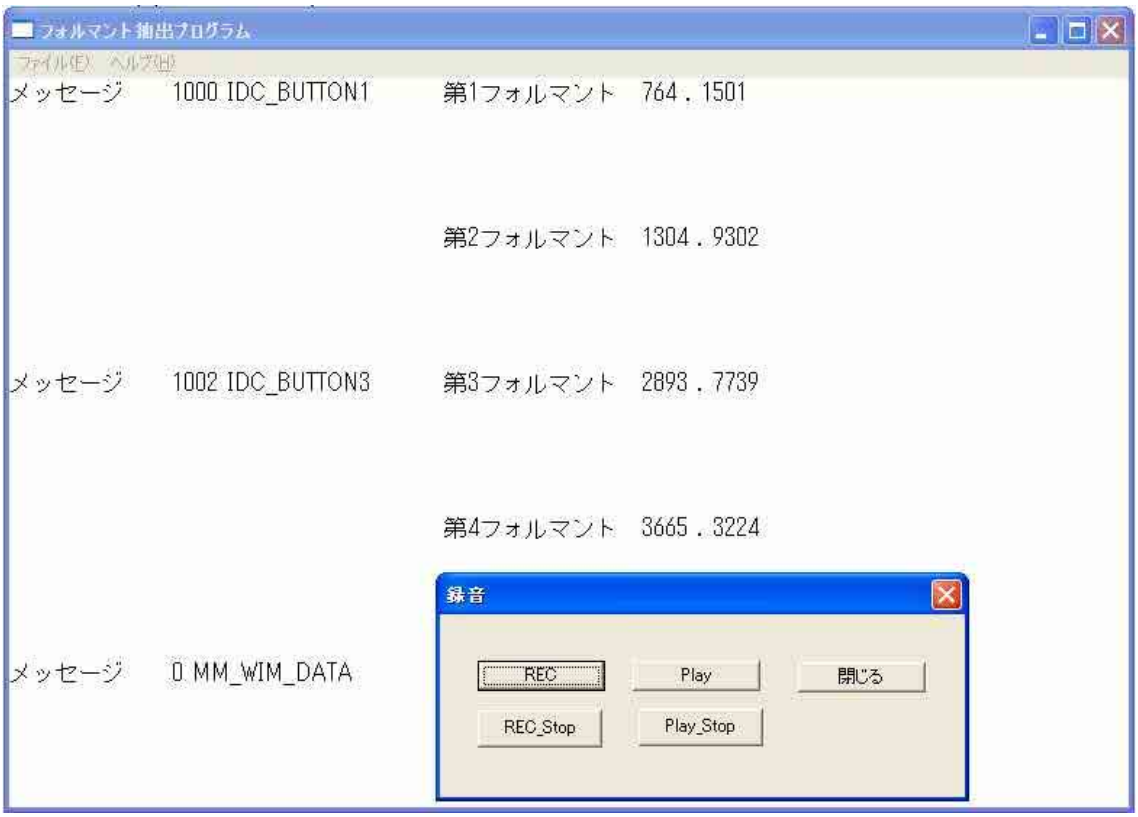

Fig.6.4　フォルマント抽出プログラム実行画面

6.5 おわりに

この章ではリアルタイムフォルマント調整システムの「フォルマント抽出プログラム」について述べた． 線形予測法を用いることで，MATLAB のプログラムとＣ言語によるプログラムとも音声データからフォルマントを抽出することができた．また，目標である「音声は WAVE ファイルなどの音声ファイルに保存するのではなく，計算機内の一時メモリに保存されたデータを計算に使用する」ことも達成した．

プログラムでは，音声の録音・再生をする関数を使いやすく記述した．そのため，今後声のピッチの取得など，音声データを必要する際には，作成したプログラムの音声データ取得部分を利用することで，効率よくプログラムを作成できるだろう．

第7章 Singer Robot とのリンク

7.1 はじめに

第 5 章でフォルマント調整プログラムを，第 6 章でフォルマント抽出プログラムを作成した．次に，この 2 つのプログラムと Singer Robot のコントロールプログラムとを組み合わせる．そのためには，プログラムにおける各パラメータを, 実際の使用する単位に返還する必要がある. 本研究では, 調整ベクトル$[\Delta F_1, \Delta F_2]^T$を実際のフォルマントに対応させるために比例定数 a を用いる．また，プログラムで使用する舌の高さ $\boldsymbol{x}$ を，コンピュータから出力電圧に対応させる．最終的に，すべてのプログラムを使い，リアルタイムフォルマント調整システムを 1 つのアプリケーションとして完成させることを目指す.

この章では，まず第 5 章で説明したフォルマント算出式（式(5-2)）の比例定数の選定をおこなう. その次に, 電圧を出力する関数 DaOutputDA への入力値と, 舌の高さの関係を示す変換式を導出する．最後に，本研究で開発したリアルタイムフォルマント調整システムのプログラムを紹介する.

7.2 比例定数の選定

7.2.1 比例定数

第 5 章で声道の形状（舌の高さ）からフォルマントを求める式を提案した. 調整プログラムがこの式(5-2)を基に考えた調整アルゴリズムを用いている．しかし，第 5 章でも触れたように，運動エネルギーの変化量 ΔW_n はフォルマントの変化量 ΔF_n と比例関係にあることまではわかっているが，その比例定数は決まっていないため，このままでは実機につなげることができない．第 5 章ではシミュレーションのみをおこなっていたため，比例定数 a を算出しなかった. 本研究の目的はリアルタイムフォルマント調整システムをロボットに実装することであるためには，調整ベクトル$[\Delta F_1, \Delta F_2]^T$を実際のフォルマントに対応させる必要がある．ここでは比例定数 a を算出する.

7.2.2 スプライン補間

駆動点間の舌が与えるフォルマント変化量を計算するために，すべての舌の高さを知る必要がある．第 5 章のプログラムでは各駆動点の高さしか指定していない．そこで，駆動点間の舌の高さを補間し，舌全体，つまり声道形状を忠実に再現しフォルマントを計算することを目指す．補間する方法はいくつかあるが，今回は３次スプライン補間を使用することにした．スプライン補間は，補間する領域をデータ間隔[x_i，x_{i+1}]に区切り，その近傍の値を使い近似の多項式で近似することを考える方法である．舌の形状（高さ）を補間するため，データ点となる駆動点の高さを含む関数を求める．そのときの補間する関数が３次関数を使うため，３次スプライン補間と呼ばれている．
３次スプライン補間でしようする関数を示す．

$$S_j(x) = a_j(x-x_j)^3 + b_j(x-x_j)^2 + c_j(x-x_j) + d_j \qquad (7\text{-}1)$$

(j = 0, 1, 2,…, N-1)

この a_j，b_j，c_j，d_j を決めなくてはならないが，本報告ではその方法の説明を省略する．

MATLAB を使用し，駆動点間の舌の高さを求めた．今回，舌の形状を再現するために，データ点を声帯から 0, 5, 10, 40, 65, 90, 115, 140, 150mm の位置に設定した．その結果を Fig.7.1 に示す．3 次スプライン補間により求めた舌の形状は，実際のロボットの舌の形状をほぼ再現できている．この補間されたデータを使い，駆動点間の舌が与えるフォルマントの変化量を求め，比例定数を求める．

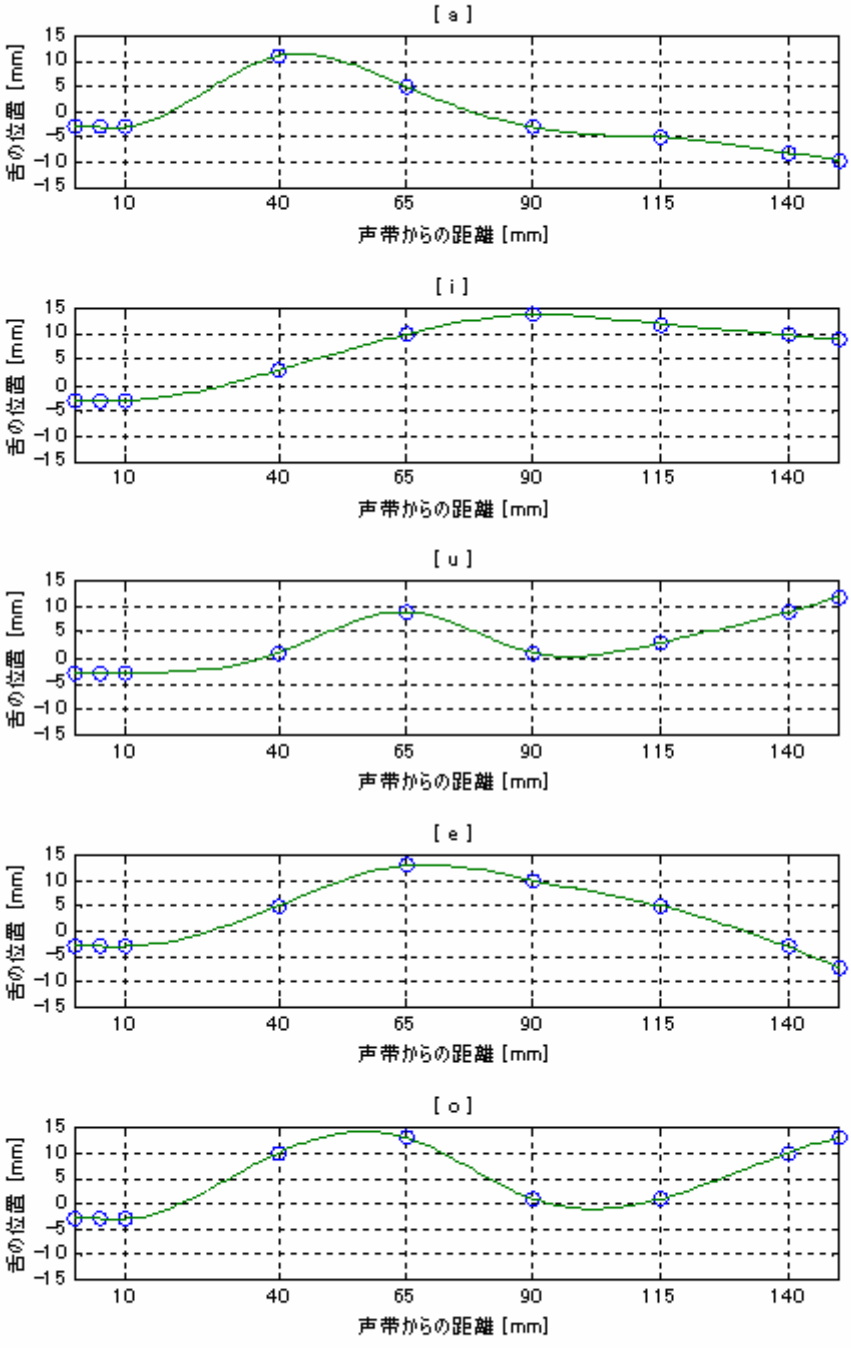

Fig.7.1　声道のスプライン補間

7.2.3 比例定数の算出

３次スプライン補間により得られた舌形状を利用し，比例定数を算出する．比例定数の算出方法は次のとおりである．まず，実機に日本語母音[a][i][u][e][o]を発声させる．そのときの実際の駆動点の高さと発声しているフォルマントも測定する．そして，MATLAB を使用し，測定した駆動点の高さの値をデータ点としてスプライン補間を行なう．今までは駆動点５箇所しか計算に入れていなかったが，スプライン補間により舌全体の高さが分かっているため，舌全体のフォルマントの影響を計算することができる．本研究では 0.1mm 刻み 1500 箇所で式(5-2)の計算を行なう．その計算により得られたフォルマント変化量と，実際のフォルマントを比較し，比例定数を決定する．

比例定数を算出するためには，フォルマントを測定するときの精度が重要になる．特に実機の気密性に注意しなければならない．フォルマントは式（5-2）をみて分かるとおり，声道の断面積 ΔA の影響を受ける．声道の気密性がない，人工舌と外枠に隙間があるということは，声道の断面積が広がっていると言うことに繋がる．正確なフォルマントを測定するには気密性を上げることが必要となる．そこで，今回は気密性を挙げるために，人工舌の側面に飴ゴムを貼り付け，人工舌と声道のアクリル板との隙間をなくして，フォルマントを測定することにした．

Fig.7.1　飴ゴムによる気密処理をおこなった Singer Robot

飴ゴムにより気密性を高めることで，安定したフォルマントを得ることができた．測定したフォルマント変化量と式（5-2）により求めたフォルマント変化量を比較した結果，第 1 フォルマントの比例定数を 0.023，第 2 フォルマントの比例定数を 0.12 とした．比例定数の単位は[Hz/mm]である．

算出した比例定数の妥当性を確認する．算出した比例定数を使い，式(5-2)から求めたフォルマント理論値と，ロボットが発声するフォルマントの測定値とを比較する．フォルマントを計算・測定した結果を Table 7.1 に示す．表の測定値が実際のフォルマントで，理論値が計算によって得られたフォルマントである．フォルマントを抽出する際，少なからず誤差が出るため，測定値と理論値の誤差が 50Hz までなら許容範囲とする．表を見ると，第 1 フォルマントについては測定値と理論値の差が 50Hz 以内に収まっていることが分かる．また，第 2 フォルマントについては，[a]，[i]では差が大きいが，[u]，[e]，[o]では差が 50Hz 以内に収まっていることが分かる．

すべてのフォルマントにおいて誤差を小さくすることができなかったが，今回算出した比例定数の妥当性は高いと言えるだろう．また，式（5-2）と比例定数によってフォルマントを計算できることが確認できた．

Table 7.1 フォルマント測定値と理論値の比較

		Ｆ１[Hz]	Ｆ２[Hz]
a	測定値	641	1346
	理論値	660	1194
i	測定値	319	1915
	理論値	323	1736
u	測定値	440	1406
	理論値	396	1455
e	測定値	473	1728
	理論値	450	1721
o	測定値	439	1092
	理論値	464	1107

7.3 DaOutputDA 入力値の変換

調整プログラムと Singer Robot のコントローラをリンクさせる際, 電圧を出力する関数 DaOutputDA への入力値と，舌の高さの関係を明確にする必要がある．なぜなら，調整プログラムでは舌の高さを[mm]で計算しているが，DaOutputDA への入力値は[mm]ではなく，分解能に対応した値になっている．使用している DA ボードの分解能は 12 ビットで, DaOutputDA に 0 という値を入力すると０Ｖ，4095 を入力すると５Ｖの電圧が出力される．一方，駆動点は DC モータの回転運動を，モータにつながるシャフトに巻かれたワイヤーによって直線運動に変換している．ＤＣモータの回転量はポテンショメータを使い検出する．ポテンショメータは有効電気的回転角度が 1080 ° で，駆動点のシャフトの直径は 5mm となっている．Fig.7.2 に駆動部の概念図を示す．

以上のことを踏まえると，DaOutputDA への入力値を x_{DA}，舌の高さを x，シャフトの直径を R とすると，x_{DA} から x に変換する式は

$$\begin{aligned} x &= x_{DA} \times \frac{5}{4095} \times \frac{1080}{5} \times \frac{\pi R}{360} \\ &= \frac{\pi}{273} \cdot x_{DA} \end{aligned} \tag{7-1}$$

となる．また，逆に x から x_{DA} に変換する式は

$$x_{DA} = \frac{273}{\pi} \cdot x \tag{7-2}$$

である．

プログラムでは初期位置として入力されている x_{DA} を式(7-1)で x に変換し，フォルマントの調整をおこなう．算出された舌の高さ x を式(7-2)で x_{DA} に変換し，DaOutputDA によってロボットへ電圧を出力する．

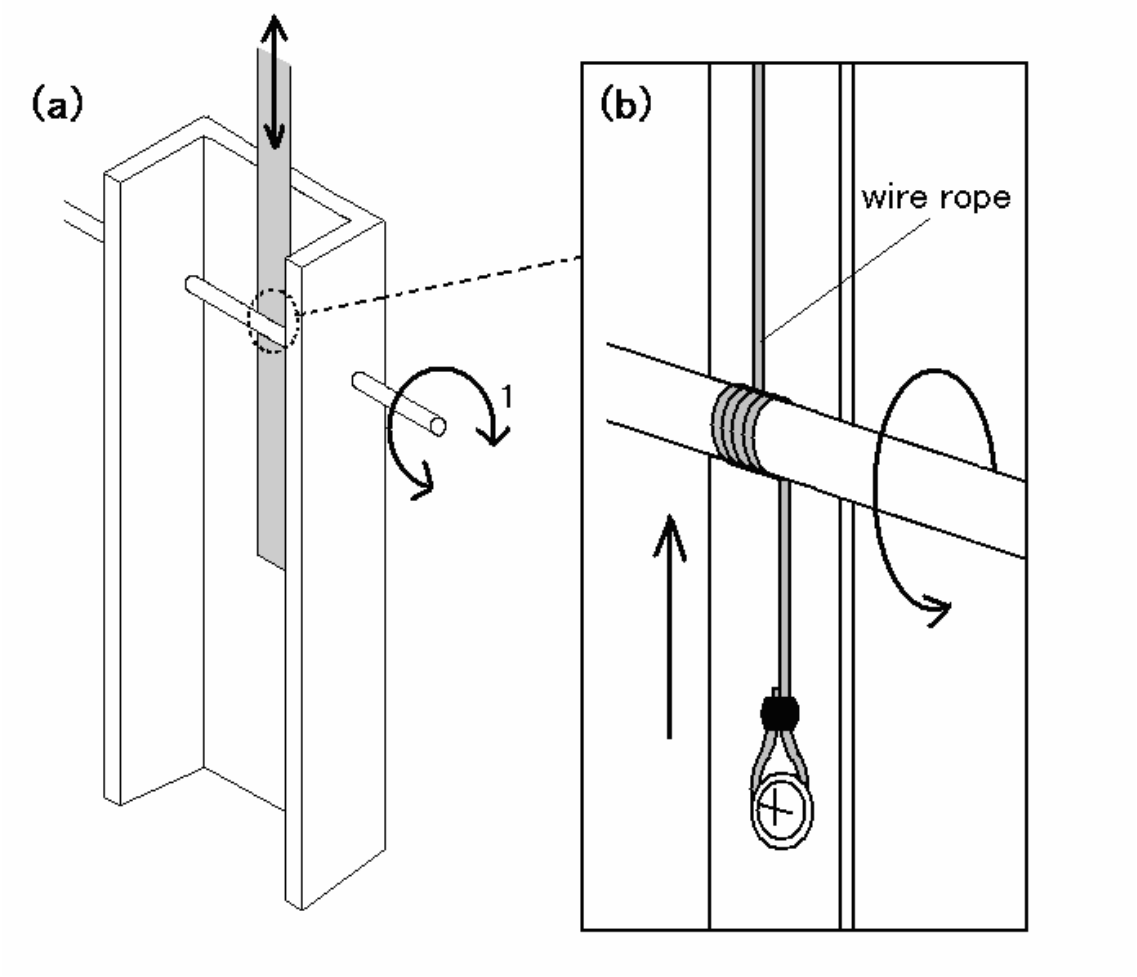

Fig.7.2　直動の仕組みの概念図（参考文献[1]）

7.4 Windows アプリケーション

比例定数 a と DaOutputDA 入力値の変換式を使い, フォルマント調整プログラムとフォルマント抽出プログラム，Singer Robot のコントロールプログラムをリンクさせる．作成したプログラムの実行結果を Fig.7.3 に示す．

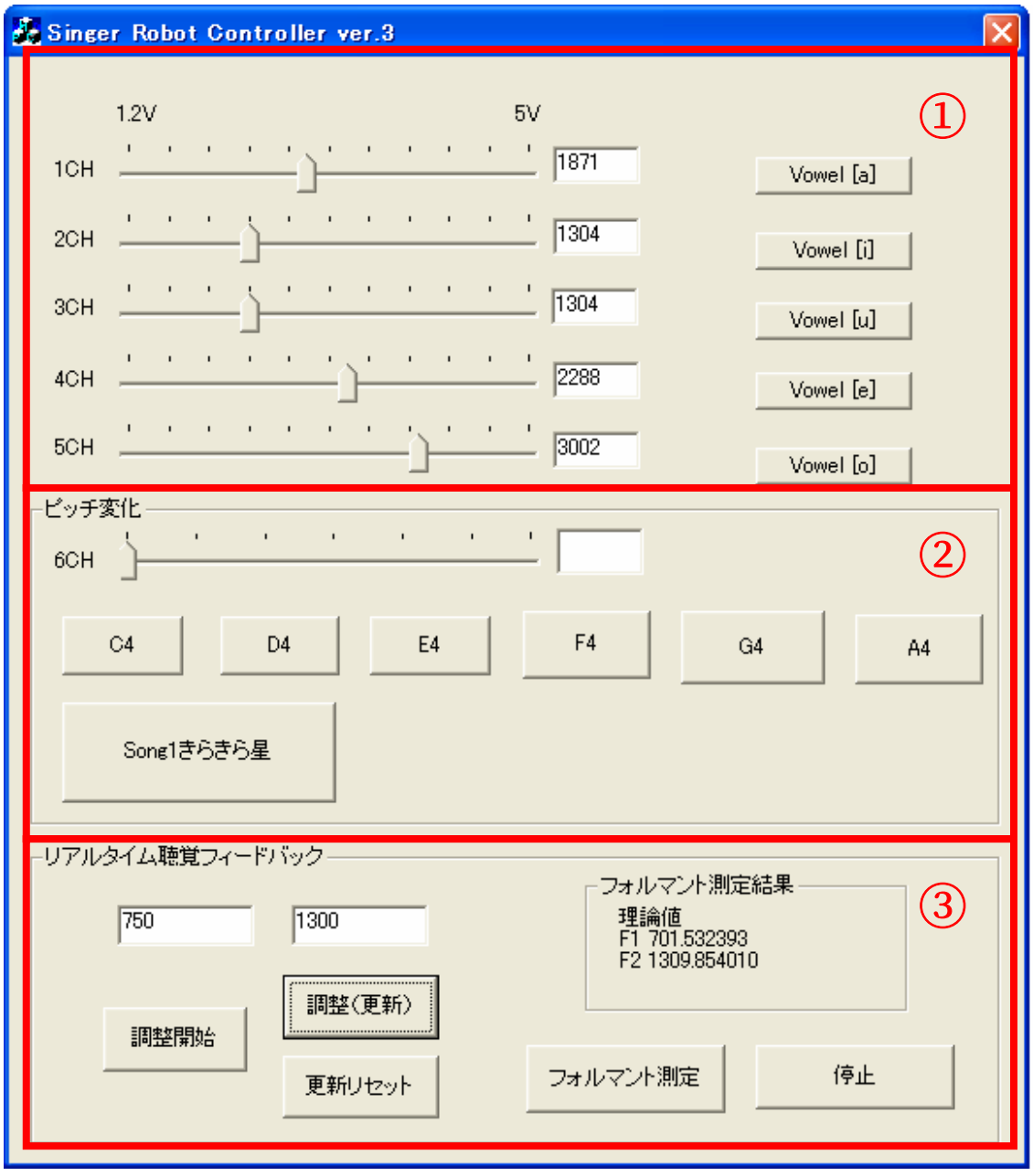

Fig.7.3　リアルタイムフォルマント調整システム

①Singer Robot コントローラ

ロボットの各駆動点における舌の高さを制御する箇所である．左のスライダを動かすと対応した駆動点のアクチュエータが動作する．右の 5 つのボタンは日本語母音[a][i][u][e][o]に対応しており，各母音を発声するときの声道形状を再現することができる．

②ピッチ変化

原音発生機構に取り付けられたピッチ変化機構を制御する箇所である．本研究では使用しない箇所であるため，説明は省略する．詳しくは参考文献[1][3]を見てほしい．

③リアルタイム聴覚フィードバック

本研究で開発したリアルタイムフォルマント調整システムをおこなう箇所である．「調整開始」ボタンを押すと，フォルマント調整を開始する．まず，ボタンの上にある 2 つのエディットボックスから目標フォルマントを取得する．左のエディットボックスが F_1，右が F_2 に対応している．次に，マイクから音声データを取得し，フォルマントを抽出する．そして，目標フォルマントと抽出したフォルマントとの誤差から舌の高さの変化量を算出し，スライドバーを移動させる．ただし，計算結果の保存はおこなっていない．「調整（更新）」ボタンは，「調整開始」ボタンと同じようにフォルマント調整をおこなうのだが，このボタンではコントローラのボタンに記憶されているデータの書き換えをおこなう．これにより，変更した舌の高さを保存しておくことができる．「更新リセット」ボタンは，「調整（更新）」ボタンにより更新されたデータを初期状態にもどすためにある．「フォルマント測定」ボタンを押すことで，マイクから入力した音声からフォルマントを抽出することができる．フォルマントの抽出結果はボタンの上に生じされる．「停止」ボタンにより，フォルマントの測定を終了する．

7.5 おわりに

この章では，フォルマント調整プログラムとフォルマント抽出プログラム，Singer Robot のコントロールプログラムの 3 つのプログラムを繋げて，リアルタイムフォルマント調整システムを完成させた．

各プログラムをリンクさせるためには，比例定数と DaOutputDA 入力値の変換式が必要であった．この 2 つの項目はプログラムとロボットをつなげる上で重要な要素となる．今後，ロボットを改良する場合には，比例定数の再選定が必要である．また，新しい駆動部の開発により，DaOutputDA 入力値の変換式もロボットに対応した式に変更することになる．

第8章 結論

本研究では，ロボットが発声した母音がより的確なフォルマントを持つようにするため，フォルマントの調整システムを開発した．そのシステムにより，フォルマントのリアルタイムでの再調整を実現することができた．

システムは，「フォルマント調整プログラム」と「フォルマント抽出プログラム」に分けて作成をおこなってきた．「フォルマント調整プログラム」では F_1-F_2 分布図における調整ベクトル$[\Delta F_1, \Delta F_2]^T$という考え方から導出したフォルマント算出式を使い，その逆問題を解くという新しいアルゴリズムを考案した．また，「フォルマント抽出プログラム」では線形予測法を用いることで，マイクから取得した音声データからフォルマントを抽出することができた．

本研究で開発したシステムによりリアルタイムでフォルマントを調整できるようになったが，このシステムは Singer Robot の状態・精度に強く影響を受けるという問題がある．例えば，声道の気密性や各駆動点の高さの追従性などである．声道の気密性が悪いと，Singer Robot が声道形状にあったフォルマントを発声することができず，システムが計算に必要な実際のフォルマントを得ることができなくなる．また，各駆動点において，人工舌の弾性力によりスライダが目標の高さまで動かない場合があり，これではシステムの計算結果を的確に反映することができない．両方ともフォルマントの調整が正確にできなくなる原因となる. そのため, 本研究で開発したシステムを有効に活用するために, Singer Robot 自体の改良が必要になってくる．

謝辞

本研究をおこなうにあたり，終始ご指導していただきました法政大学工学部機械工学科，高島俊教授に深く感謝をあらわすとともに御礼申し上げます.

また，本論文を作成するにあたり，同期生として切磋琢磨した法政大学工学研究科機械工学専攻修士課程の渋谷純也君，鈴木洋平君，関隆行君に心より感謝申し上げます.

さらに在学中にご指導していただきました法政大学大学院機械工学専攻修士課程卒業生の山本健太郎氏, 本研究を共同して行った松江健司君, 高橋重治君, 実験などに協力していただいた法政大学大学院工学研究科機械工学専攻修士課程 1 年および法政大学工学部機械工学科 4 年の高島研究室の皆様に心から感謝申し上げます.

2008 年 2 月 20 日

中野　陽介

発表論文

1） 中野陽介，高島俊，山本健太郎，“ 人工喉頭を用いた Singer Robot の研究 ‐母音フォルマントの最適調整‐ ”，日本機械学会ロボティクス・メカトロニクス講演会 ‘ 06 講演論文集

2） 中野陽介，高島俊，“ 人工喉頭を用いた Singer Robot の研究 ‐母音フォルマントのリアルタイム最適調整‐ ”，日本機械学会ロボティクス・メカトロニクス講演会 ‘ 07 講演論文集

3） Yosuke Nakano, Suguru Takashima, “Study on a Singer Robot using Artificial Larynx: Optimal Real-time Adjustment of Vowel Formants”, ICAR2007

参考文献

書籍

[1] 和田恭平, “ 人工喉頭を用いた Singer Robot の基礎研究 ”, 法政大学大学院修士論文, 2004.

[2] 高島俊, 和田恭平, “ 人工喉頭を用いた Singer Robot の基礎研究　－母音を連続生成する調音機構－ ”, 日本機械学会ロボティクス・メカトロニクス講演会‘05 講演論文集, 2005.

[3] 山本健太郎, “ 人工喉頭を用いた Singer Robot の研究　- 喉頭原音ピッチの操作による発声音の制御 - ”, 法政大学大学院修士論文, 2006.

[4] 中野栄二, “ ロボット工学入門 ”, オーム社, 1983.

[5] T.Chiba and K.Kaziyama, “ The Vowel –Its Nature and Structure－ ”, Kaiseikan, 1942., （訳本, 岩波書店, 2003.）

[6] Kenneth N.Stiven, “ Acoustic Phonetics ”, The MIT Press, 1998.

[7] 古井　他, “ 音響・音声工学 ”, 近代科学社, 1992.

[8] 三浦　他, “ 新版 聴覚と音声 ”, 電子情報通信学会, 1980.

[9] Ray D.Kent, Charles Read, “ 音声の音響分析 ”, 海文堂, 1996.

[10] 板橋秀一, “ 音声工学 ”, 森北出版株式会社, 2005.

[11] Lawrence Rabiner,Biing-Hwang Juang, “ 音声認識の基礎（上）”, NTT アドバンステクノロジ株式会社, 1995.

[12] 服部四郎, “ 音声学 ”, 岩波書店, 1945.

[13] 中田和男, “ 改訂　音声 ”, コロナ社, 1995.

[14] 薄井, 荒井, 村原, “ 母音生成に対する音響教育を目的とした声道模型の作成 ”, 日本音響学会講演論文集, 2001.

[15] 長松昭男, “ モード解析入門 ”, コロナ社, 1993.

[16] 小林一行, “ 最新 MATLAB ハンドブック ”, 秀和システム, 2004.

[17] 喜納秀明, “ システムの最適理論と最適化 ”, コロナ社, 1987.

[18] (株)アンク, “ Ｃ言語が好きになる 9 つの扉　Ｃの絵本 ”, 翔泳社, 2002.

[19] 小高知宏, “ はじめて学ぶＣ言語[システム構築編] ”, ナツメ社, 2003.

[20] システム制御情報学会, “ 遺伝アルゴリズムと最適化 ”, 朝倉書店, 1998.

[21] 白石洋一, “ 組合せ最適化アルゴリズムの最新手法 -基礎から工学応用まで- ”, 丸善株式会社, 2002.

[22] 田辺義和, “ Windows サウンドプログラミング ”, 翔泳社, 2001.

HomePage

[1] SoftComputing lab. http://scl.m-kb.net/ga-2.shtml
[2] 沖電気 http://www.oki.com/jp/RDG/JIS/oto/speech/
[3] Tomy's Home Page http://www5.airnet.ne.jp/tomy/cpro/csource.htm
[4] WisdomSoft http://wisdom.sakura.ne.jp/system/winapi/index.html
[5] Codean http://www.kab-studio.biz/Programing/Codian/

付録

作成したプログラムのソースを紹介する．本研究では，本文で説明したプログラムを今まで作成されてきた Singer Robot のコントロールプログラムに追加する形で作成している．このプログラムは Microsoft Visual C++6.0 の MFC を用いて書かれたものである．

ソースファイルは

sldr5.cpp
sldr5Dlg.cpp
StdAfx.cpp
Adjustment.cpp
complex.cpp
Extraction.cpp
GetMessageHook.cpp
LPC.cpp
poly.cpp
SingerRobotParameter.cpp
sort.cpp
spline.cpp
waveInOut.cpp

ヘッダーファイルは

Resource.h
sldr5.h
sldr5Dlg.h
StdAfx.h
Adjustment.h
constant.h
Extraction.h
GetMessageHook.h
LPC.h
SingerRobotParameter.h

sort.h
spline.h
sslib.h
waveInOut.h

からなる.

すべてのファイルを記載すると膨大な量になるため，プログラムで重要なファイルのみ付録として紹介する．第 5 章の「フォルマント調整プログラム」と第 6 章の「フォルマント抽出プログラム」のソースを以下に示す.

「フォルマント調整プログラム」
Adjustment.cpp

「フォルマント抽出プログラム」
Extraction.cpp

Adjustment.cpp

```
/*********************************************************/
/*              母音フォルマント最適調整プログラム                */
/*********************************************************/

//このプログラムは母音フォルマントの最適調整プログラムです.
//最適化にはタブーサーチを使用

#include "stdafx.h"
#include "Adjustment.h"

#ifndef _MYHEADER_
#define _MYHEADER_

#define Fs 11025                        //  サンプリング周波数

#include"constant.h"
#include"sslib.h"
#include"sort.h"
#include"Adjustment.h"
#include"spline.h"
#include"waveInOut.h"
#include"Extraction.h"
#include"LPC.h"
#include "GetMessageHook.h"

#endif //_MYHEADER_

/*グローバル変数の宣言*/
double targetY[2],initialY[2];
double *X;                  //計算用ポインタ   いらないかも


/*******************/
/*   Adjust()関数    */
/*******************/
double Adjust(double x0[5],                                     //舌の位置
                                        double realF[2],                        //フォルマント実測値
                                        double targetF[2],      //フォルマント目標値
                                        double returnDF[2])     //フォルマント変化量の理論値(計算結果の受け取り)
{
        int i,j;
        double x[5],s[5],vtF[2],deltaY[2];
//      unsigned long t0,t1;                    //時間測定用


        /*****************************************************************/
        //formant 関数用の変数
        //メモリ確保
        //未実装
        /*****************************************************************/


        //  声道の共鳴周波数
/*      double CV = 350;                                        //共鳴周波数計算
        for(i=0;i<2;++i){
                vtF[i]=(2*(i+1)-1)*CV/(4*VocalTract_L)*1000;            /*CV(m/s)->(cm/s)   VocalTract_L(mm)->(cm)
        }
*/
        vtF[0] = 502;
        vtF[1] = 1475;

        X=&x0[0];                                               /*ポインタ*/
```

```
	//	初期位置でのフォルマントの計算

	for(i=0;i<2;++i){
		initialY[i]=formant(i,X)+vtF[i];			//initialY[i]:初期位置でのフォルマント
	};


	/**フォルマント測定値と目標値の差を求める**/
		for(i=0 ; i<2 ; i++){
			deltaY[i] = targetF[i] - realF[i];			//deltaY[i]：目標変化量
		}

		for(i=0;i<2;++i){
			targetY[i]=deltaY[i]+initialY[i];	//
		}

	/*初期解の決定*/
		for(i=0;i<5;++i){
			x[i]=rndn(1000)/1000.*rui(-1,(int)rndn(2))*20;
		};

//			t0=clock();				//////////*時間測定開始*//////////

	/*タブーサーチ*/
		for(i=0;i<5;++i){
			s[i]=x[i];
		}
		TS(x,s);

//			t1=clock();				//////////*時間測定終了*//////////
//			printf("計算時間	%f 秒¥n",(t1-t0)/1000.0);


		for(j=0;j<5;++j){
			x0[j]=x0[j]+x[j];
		}
		X=&x0[0];

	/*****************************************************************/
	//formant 関数用の変数
	//メモリ解放
	//未実装
	/*****************************************************************/

	//	フォルマント理論値を返す
	for(i=0;i<2;++i){
		returnDF[i]=formant(i,x)+realF[i];
	};
	return 0;

}



/***************************/
/*	タブーサーチプログラム	*/
/***************************/
void TS(double x[5],			//舌の位置
		double s[5])			//計算用
{
	int i, j, k=0, taboo[15], maxtaboo, c, R=7;
	double T[15][5], n[5], calcfit_s, calcfit_x = NULL;
	char taboo_count = 0;

/*タブーリストの初期化*/
	for(i=0;i<15;++i){
		for(j=0;j<5;++j){
			T[i][j]=NULL;
		}
```

```
	};

	//タブーサーチのパラメータを変更 2008/02/03
	for(c=0;c<100;++c){			//200->100
		if(c%2==1){
			R=15;
		}else{
			R=5;			/*タブーリストのサイズ変更*/
		}

		seach(s,n);					/*試行解*/

	/*タブーリストとの比較*/
		maxtaboo=0;
		for(i=0;i<R;++i){
			taboo[i]=0;
		}
		for(i=0;i<R;++i){
			for(j=0;j<5;++j){
				if((T[i][j]-0.05)<s[j] && s[j]<(T[i][j]+0.05)){
								taboo[i]=taboo[i]+1;
				}
			}
			if(maxtaboo<taboo[i]){
				maxtaboo=taboo[i];
			}
		}

		//15 回同じ値が出た場合，計算をやめる
		if((x[0] == s[0]) && (x[1] == s[1]) && (x[2] == s[2])
			&& (x[3] == s[3]) && x[4] == s[4]){
			taboo_count++;
		}else{
			//計算回数を減らすようにしてあります
			calcfit_s = calcfit(s);
			if(calcfit_x == NULL)
				calcfit_x = calcfit(x);

			if(maxtaboo<2){
				for(i=0;i<5;++i){
					x[i]=s[i];
				}
				calcfit_x = calcfit_s;
			}else if(calcfit_s < calcfit_x){
				for(i=0;i<5;++i){
					x[i]=s[i];
				}
				calcfit_x = calcfit_s;
			};
		}
		if(taboo_count >= 15)
			break;

	/*タブーリストの更新*/
		if(k>=15){
			k=0;
		}
		for(j=0;j<5;++j){
			T[k][j]=s[j];
		}
		k=k+1;
	};
}


/**********************/
/*	局所探索	*/
/**********************/
void seach(double s[5],		//舌の位置
		double n[5])			//計算用
```

```
{
	int i,j,c,FF=0;
	double d,fugo;

	//パラメータの変更 2008/02/03
	for(c=0;c<30;++c){ //15->30
	/*解候補の選出*/
		for(j=0;j<5;++j){
			fugo=rui(-1,(int)rndn(10));
			d=0.001*(int)rndn(100);
			n[j]=s[j]+d*fugo ;
			if(*(X+j)+n[j]>20){ // 最 大 値 の 指 定
	//<-駆動範囲に対応
				n[j]=n[j]-2*d;
			}else if(*(X+j)+n[j] < 0){ //最小値の指定
				n[j]=n[j]+2*d;
			}

		};

	/*評価のよい解を選択*/
		if(FF=0){
			for(i=0;i<5;++i){
				s[i]=n[i];
				FF=1;
			}
		}else if(calcfit(s) > calcfit(n)){
			for(i=0;i<5;++i){
				s[i]=n[i];
			}
		}
	}
}


/************************/
/*  評価値(適応度)の計算  */
/************************/
double calcfit(double x[5])			//舌の位置
{
//評価値の計算です．目的関数を最小化します.

	int i,j;
	double L[2],yy[2],LL,xx=0,K=0,tmp[5]={0,0,0,0,0};

	extern double targetY[2],*X,initialY[2];

	for(i=0;i<2;++i){
		yy[i]=formant(i,x)+initialY[i];		/*フォルマントを計算　Y:初期位置のフォルマント*/
	};
	for(j=0;j<5;++j){

		xx=xx+zi(x[j]);			/*Ｘの動きを小さくする*/

		tmp[j]=*(X+j)+x[j]; /* κ の 計 算 */
	//<-駆動範囲に対応
		if(tmp[j] > 19){ // 最 大 値 の 指 定
	20-1
			tmp[j]=zi((tmp[j]-19)*100);
		}else if(tmp[j] < 0){ //最小値の指定 0
			tmp[j]=zi((0-tmp[j])*100);
		}else{
			tmp[j]=0;
		}

		K=K+tmp[j];
	};
//	L[i] = zi(targetY[i]-yy[i])+10*xx+K; /*+10*zi(y[i]-realF[i])*///  realF（フォルマントの実測値）を考慮するか検
```

```
討中
		LL = 100*(zi(targetY[0]-yy[0])+zi(targetY[1]-yy[1]))+10*xx+10*K;

//		LL=L[0]+L[1];
		return LL ;
}


/******************/
/*n 未満の整数乱数  */
/******************/
double rndn(double limit)		//ランダムの最大値
{
		double res=limit ;
		static int flag;

		if(flag == 0){
			srand((unsigned int)time(NULL));
			flag = 1;
		};

		while(res>=limit){				/*0 から limit の間になるまで繰り返し*/
			res=rand() / (double)RAND_MAX * limit ;
		}

		return res ;
}


/******************/
/*    自乗		*/
/******************/
double zi(double x)
{
		double y;
		y=x*x;
		return(y);
}


/******************/
/*     累乗		*/
/******************/
int rui(int x,int b)
{
		int i,y=1;
		for(i=0;i<b;++i){
			y =y*x;
		}
		return(y);
}


/*******************************/
/*  スライド間にある測定点の高さ*/
/*******************************/
double high(double xh1,double xh2, double theta)
{
		/* θ は x(p)を 0 度，x(p+1)を 180 度とする*/
		double C,hight;

		C=cos(theta/180*M_PI);			/*度からラジアンに変換して，コサインの値を出す*/
		hight = xh1*(1+C)/2+xh2*(1-C)/2;

		return(hight);
}


/*****************************************/
/*    スライド間にある測定点の声帯からの距離    */
```

```
/*****************************************/
double dis(int m,double xl1,double xl2)
{
	double kyori;

	kyori = xl1*(4-m)/4+xl2*m/4;
	return(kyori);
}


/*********************************/
/*   ⊿Fn の計算                  */
/*********************************/
double Df(int n,double l)
{
	double delta;

	delta = cos((2*n-1)*M_PI*l/VocalTract_L);
	return(delta);
}


/*********************************/
/*    フォルマント変化量の計算    */
/*********************************/
double formant(int n,double x[5])
{
	//   スプラインの適応  2007/11/23
	int for_N;
	double F=0;

	double *h, *b, *d, *g, *u, *r, *L, *for_x, sp, x1, y1;

	double alpha[] = {0.023, 0.12};                                    // 比例定数


	// 設定
	for_N = 10;                                                  // 区間の数
	h = new double [for_N];                  // 計算領域
	b = new double [for_N];
	d = new double [for_N];
	g = new double [for_N];
	u = new double [for_N];
	r = new double [for_N+1];

	for_x = new double [for_N+1];
	L = new double [for_N+1];

	// 声帯からの距離
	L[0] = 0.0;
	L[1] = 5.0;
	L[2] = 10.0;
	L[3] = 15.0;
	L[4] = 45.0;
	L[5] = 70.0;
	L[6] = 95.0;
	L[7] = 120.0;
	L[8] = 145.0;
	L[9] = 155.0;

	// データの別変数に再格納
	for_x[0] = 0.0;
	for_x[1] = 2.0;
	for_x[2] = 2.0;
	for_x[3] = 2.0;
	for_x[4] = x[0];
	for_x[5] = x[1];
	for_x[6] = x[2];
	for_x[7] = x[3];
	for_x[8] = x[4];
```

```
	for_x[9] = x[4]+(x[4]-x[3])/2.0;

	// 実行と出力
	x1 = 0.0;
	sp = 0.1;
	while (x1 <= 155.0) {
		y1 = spline(for_N, L, for_x, x1, h, b, d, g, u, r);

		F = F+ Df(n+1, x1) * y1 * alpha[n];

		x1 += sp;
	}

	delete [] h;
	delete [] b;
	delete [] d;
	delete [] g;
	delete [] u;
	delete [] r;
	delete [] for_x;
	delete [] L;

	return(F);
}
```

Extraction.cpp

```
//フォルマント抽出プログラム//
//フォルマント抽出プログラムのソースファイルです

#include "stdafx.h"
#include "Extraction.h"

#ifndef _MYHEADER_
#define _MYHEADER_

#define Fs 11025                        //  サンプリング周波数

#include"constant.h"
#include"sslib.h"
#include"sort.h"
#include"Adjustment.h"
#include"spline.h"
#include"waveInOut.h"
#include"Extraction.h"
#include"LPC.h"
#include "GetMessageHook.h"

#endif //_MYHEADER_


void GetFormant(LPSTR lpData, double Formant[Formant_Max])
{
        ////////////  フォルマント抽出の計算をします
        int i;
        double Xp[sample_N];
        double Xh[sample_N];
        Complex x[order];
        double alpha[order];
        double F_data[order];
        int F_num;



        short* asdf;
        asdf = (short*)lpData;

        double a[331];

        for(i=0;i<331;i++){
                a[i] = *(asdf+i+2000);
        }


        //  プリエンファシス回路
        for(i = 1;i < sample_N+1; i++){
                Xp[i-1] = a[i]-0.95* a[i-1];
        }

        //  窓掛け（ハミング窓）
        for(i = 0;i < sample_N; i++){
                Xh[i] = Xp[i] * ( 0.54-0.46*cos((2*M_PI*(i+1))/(sample_N-1)) );
        }

        //  線形予測法（ＬＰＣ）  確認ＯＫ
        LPC(Xh, alpha);

        //  伝達関数の極を求める  確認ＯＫ
        double alpha2[order+1];
        for(i=0;i<order;i++){
                alpha2[i+1] = alpha[i];
```

```
		}
		alpha2[0] = 1;
		bairs(alpha2, order, 1.e-6, 100, x);

		//	フォルマントの算出

		for(i = 0;i < order; i++){
			F_data[i] = 1/(2*M_PI*1/Fs)*((cln(x[i])).i);
		}
		qsort_d(F_data,order);

		//	フォルマントを抽出
		F_num = 0;
		for(i = 0;i < order; i++){
		if(0 < F_data[i]){
			Formant[F_num] = F_data[i];
			F_num++;
			if(F_num >= Formant_Max)
				break;
			}
		}
}
```