Flash コンテンツ操作のための音声認識インタフェース
A Spoken Dialogue Interface for ActionScript Software Control

松浦 健太
Matsuura Kenta
法政大学情報科学部ディジタルメディア学科
E-mail: kenta.matsuura.bg@cis.hosei.ac.jp

Abstract

Adobe Flash is a software that has spread worldwide, but it has not created plug-in that enables the voice operation. Therefore, an interface of voice interaction for Flash contents operation is proposed in this paper. It is required that contents are operated as defined as correspondence of a voice command, and a mistake in an instruction recognizing is evaded. Therefore, an ActionScript library, a program of voice activity detection, an engine of speech recognition, and a system construction is researched as the research content. The ActionScript library provides the followings. The first is a configure tool for the engine of speech recognition needed in recognizing the instruction. Next is a cool back event that starts a sequence function when the instruction is received. A module mode of free software Julius is used as the engine of speech recognition. It is feared that the noise is mistaken for command and Flash become malfunction. Therefore, the program of voice activity detection is used to prevent a noise from influencing the speech recognition. The recognition results were that a word correct answer rate went up 14% than an existing method. The result of the precision was satisfied because the fear which the noise is misidentified is a little. But, it was necessary to improve the average delay time when thinking about the practicality.

1 まえがき

Flash とはアドビシステムズが開発している動画やゲーム等を扱うための規格である．幅広いコンテンツを作成出来，また再生環境への依存度の低いことから，世界的に利用されている．したがって，Flash のプログラミング言語である ActionScript を扱えるプログラマも多い．しかし，音声認識に関して知識を持っている者はその一部である．一方で，音声は情報を伝達する手段の中で，最も古く，最もポピュラーなものである．これは，音声がインプット手段として適していることを意味している．Adobe Flash でリアルタイム音声対話インタフェース [1] 作成することが出来れば，ユーザが呪文を唱えればキャラクタが魔法を使う等，よりインタラクティブなコンテンツを作成出来るようになり，対話インタフェースの可能性を大きく広げることが出来る．

2 音声認識による Flash コンテンツ操作インタフェース

ActionScript を使用するプログラマは多いため，プラグインツールは公式のものだけでなく，自作のものもオープンソースとして出回っている．そのため，google マップ等，多様な機能を利用することが出来る．しかし，音声認識を行うものは，現状では出回っていない．これを作るにあたり次の 3 つの課題を挙げる．

まず，音声認識の知識が無いプログラマでも，音声認識インタフェースを作成出来るようにする．専門知識が必要になると，使用者は限られてしまう．これを防ぐため，音素ごとの認識精度の違いを明確にする表を提供する等の対応が挙げられる．

次に，容易にコンテンツを拡張出来るように必要な設定を簡潔にする．既存のコンテンツを音声認識出来るように拡張するときや，命令コードを追加するときに必要な処理が簡単に行えなければ，プラグインは出回りにくくなる．音声認識システムに一切設定しなくても，ActionScript で必要な命令コードを設定するだけで，音声認識システムの辞書を自動で作成する等の対応が必要である．また，音声認識システムから認識結果を受け取り，必要な関数を呼び出すため，認識結果ごとにイベントを定義する．

最後に，ActionScript プログラマが安心して音声認識インタフェースを提供出来るようにする．音声認識精度が低く，頻繁にコンテンツを誤作動させるなら音声認識を行わない方が良い．認識精度を十分に向上し，また認識時の遅延時間を短縮する必要がある．

この論文では，音声認識システムの認識精度向上，音声認識エンジンの設定及びイベントを定義する ActionScript ライブラリを作成する．

3 Flash コンテンツ音声操作システムの概要

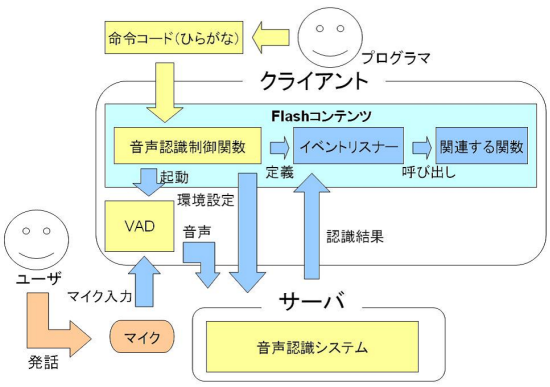

図 1. Flash コンテンツ音声操作システム

プログラマが命令コードの一覧を音声認識制御関数に渡すことにより，イベントリスナーを定義，音声区間検出プログラム (VAD) を起動，音声認識システムサーバに接続し命令コードの一覧を認識出来るようするための環境設定を行う．VAD はマイクから音声を検出すると音声認識システムに音データを送り，音声認識させる．認識結果は ActionScript に渡され，イベントリスナーから認識結果に必要な関数を呼び出す．

Supervisor: Prof. Katunobu Itou

3.1 音声認識のための ActionScript 拡張ライブラリ

上記のシステムを構築する上で，ActionScript ライブラリには辞書設定，VAD 起動，イベント定義の 3 つの機能を持たせる．

辞書とは認識したい命令コードを定義するもので，ここに登録することで音声認識システムはその命令を認識することが出来るようになる．この設定は，文字列型の配列に命令コードの読みを「ひらがな」で入れ，関数に渡すことで，自動に配列を音声認識システムの辞書のフォーマットに書き換え，音声認識システムに渡す．

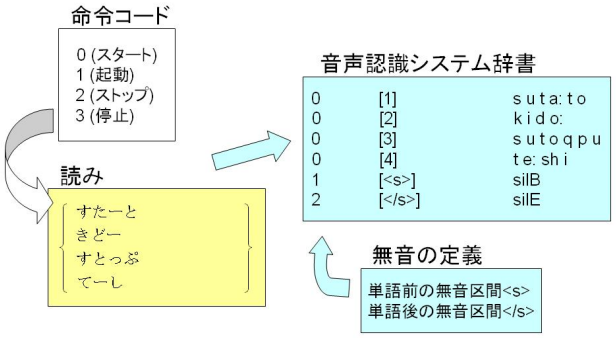

図 2. 配列を辞書に変換

また，音声認識システムから命令内容を受け取ったとき，認識結果に応じたイベントをコールバックする．既存のイベントリスナーはソケットからデータを受け取った事を伝えるイベントを発生出来る．しかし，これではどんな認識結果が来て，どの関数を処理すればいいのか分からない．よって，認識結果ごとに関連した関数を呼び出すためのイベントを定義する必要がある．

3.2 音声認識システム

音声認識システムは音声を文字に変換するもので，本稿ではフリーソフト julius[2] を用いる．julius は，音声認識システムの開発・研究のためのオープンソースソフトウェアで，ソースコードを含めて誰でもフリーで入手することができる高性能な汎用大語彙連続音声認識エンジンである．認識対象はマイク入力，録音済みの音声波形ファイルおよび特徴抽出したパラメータファイルに対応している．また外部クライアントと julius は，ソケットを介したサーバークライアントの形式で接続することが出来る．クライアントは認識開始・一時停止等の動作命令を送り，julius からは，認識結果や音声のトリガ情報等を受け取る．

本稿では julius をサーバとして，ActionScript を外部クライアントとして TCP 通信する．辞書を ActionScript から受けとり，その辞書を用いて音声データから命令を認識する．

4 VAD

4.1 音声認識システムの性能評価

音声認識では，二つのモデルと発音辞書を利用して，音声を文字に変換する．音響モデルは母音や子音等の発音記号ごとに，声の音響的な特徴を表現する．一方，言語モデルは，文の最初にどのような単語がよく現れるか，あるいは，ある単語の後ろには，どのような単語がよく使われるか等を表すモデルである．発音辞書は二つのモデルの架け橋となるものであり，各単語に対して発音記号が記されている．

音声認識の処理の流れを図 3 に示す．言語モデルを利用して文頭に現れやすい単語の候補をリストアップし，入力音声と照合する．次に，文頭の単語に接続し得る単語の候補を言語モデルからリストアップし，入力音声と照合する．このような処理を，入力音声の最後まで行い，認識結果を確定する．音声認識では，言語モデルを参照しながら，音響モデルを用いた照合を繰り返して，認識結果を確定していく処理を，照合（サーチ）と呼ぶ．

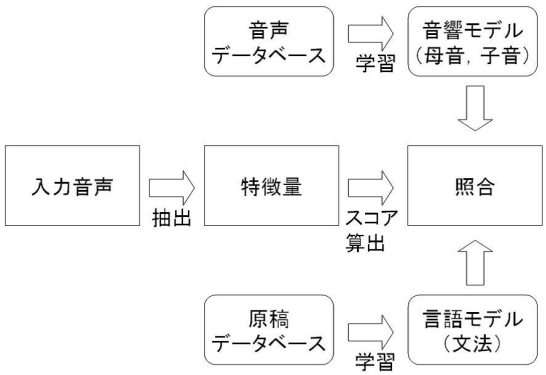

図 3. 音声認識の原理

音声認識を行う際に問題となるのが，雑音の影響である．入力音声に雑音が含まれている場合，音声だけでなく雑音の特徴量も抽出するため，認識結果に影響が出る．

音声認識での雑音対策として特徴量抽出前，音響モデルとの照合時，言語モデルとの照合時の三箇所で行われる．特徴量抽出前に行う雑音対策としては，雑音除去，音声強調，音声区間検出等が挙げられる．音響モデルとの照合時に行う雑音対策としては，音響モデルの学習の際に、クリーン音声だけでなく雑音を含む音声を用いて学習する等の方法がある．言語モデルとの照合時に行う雑音対策としては，単語と単語の間に雑音が含まれている場合の言語モデルも用意して照合する等の方法がある．

本稿では特徴量抽出する前に，雑音除去と音声区間検出を用いた雑音対策を検討する．雑音除去は音声に重なっている雑音を除去することが出来る．本稿では音声以外の音を雑音だと定め，Flash コンテンツを操作する際に得られる雑音として，衝突音や音楽，キーボード，ファン等を想定する．この雑音を除去する手法として，スペクトルサブトラクション法 (SS 法)[3]，ランニングスペクトルフィルタ (RSF)[4] を使用した．SS 法と RSF 法は定常的な雑音に効きやすい．これにより，ファン等の音を消すことが出来た．しかし，キーボード等は突発的な雑音であり，大きな効果は得られなかった．また，音楽に関してはミュージカルノイズが発生してしまい，認識精度が下がった．ミュージカルノイズが発生しないようにしようとすると，雑音が大して減少しなかった．雑音除去の影響は雑音だけでなく，音声にも生じた．音声信号の一部が欠落し，音声の特徴量に影響した．これらのことにより，本稿では雑音除去を有効な手段ではないと判断した．

そこで誤認してしまう箇所を調べたところ，音声に雑音が図 4 の (A) のように重なっている影響は少なく，大きく影響を与える雑音は (B) のように突発的に起こる雑音や，(C) のように音声の直前や直後にある雑音だと分かった．

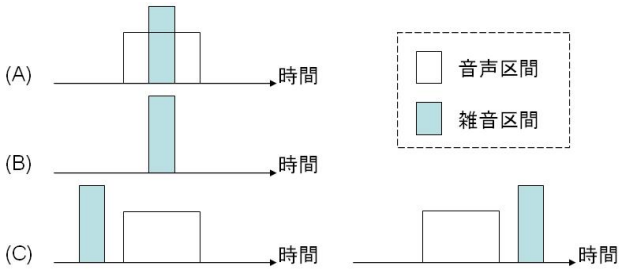

図 4. 音声認識に影響する雑音パターン

音声認識システムへの入力の場合，音声は周りの環境音よりもパワーが大きく，例え音声に雑音が重なっていたとしても特

徴量への影響が少ない．しかし，音声がない区間では雑音の音だけが特徴量として認識される．雑音が大きいと，(B) のような雑音のみでも音声として誤認することがある．また雑音が大きくなくても，(C) のように近くに音声が含まれていると，雑音＋音声もしくは音声＋雑音で特徴量を抽出し，尤度計算を行う．その結果，スコアに影響が出てしまい，認識精度が低くなる．音声区間のみを抽出し認識した結果，高い認識精度を得られた．

よって，本稿では雑音除去ではなく，VAD によって音声認識精度向上を図る．

4.2 PARADE と GMM を用いた VAD

julius には VAD 機能が備わっている．julius に内蔵されている VAD は二つの手段を用いている．一つ目は音声信号の振幅と零交差数に基づいて，音入力の開始と終了を検出する方法である．一定のレベルを超える振幅について零交差数が一定数を超えたとき，音声始端として認識処理を開始，値が一定以下になったときに音声の終端として入力を区切って 1 発話分の認識を終了する．この方法は最も簡単な方法の一つであり，計算量が少ないという利点がある．しかし，雑音の音量が大きい環境では，音声と同じように雑音も一定以上の振幅と零交差数となり音声と雑音の判別が不可能となってしまう．

二つ目はより精密に音声区間を抽出するために，GMM(Gaussian mixture model) による VAD[5] が用いられている．これは GMM に基づく音声区間検出で，音情報から特徴量を抽出して尤度を計算する．GMM による認識は倍音構造になっている雑音には頑健だが，衝突音等突発的雑音には弱い．

本稿で用いる VAD は，この GMM による VAD に加えて PARADE(periodic to a periodic component ratio-based detection) [6] を用いる．先にも述べた通り，GMM による VAD は突発的雑音に弱い．PARADE はその弱点を補うことが出来る．PARADE はスペクトルの倍音構造を調べて周期性成分と非周期性成分との比を算出し，その周期性の割合で音声区間を抽出する．PARADE は倍音構造を持つ雑音には弱いが，突発的な雑音に頑健である．

4.2.1 GMM による VAD

事前に音声データと無音データを録音し，特徴量を抽出して音声状態モデルと非音声状態モデルを学習する．そして，音声区間検出を行う際，音声状態と非音声状態の尤度を計算する．この尤度の比率が一定時間の間，閾値を超えていれば音声と判断する．尤度の計算は窓幅 110〜200ms（一つの音素が約 100ms より），ずらし幅 10〜50ms で窓がけする．学習方法には EM アルゴリズムを用いる．また，特徴量として，MFCC12 次元，ΔMFCC12 次元の計 24 次元を用いた．

4.2.2 PARADE

母音等の音声にはスペクトルに倍音構造が見られる．この倍音構造は衝突音等の雑音にはない．PARADE は観測信号がこの倍音構造を持っているかを調べることで雑音か音声かを判別する．

観測信号が，周期性成分 (基本周波数 F0 とその倍音成分から成る調波成分) と非周期性成分の和から成ると仮定する．各フレームにおける周期性成分のパワーと非周期性成分のパワーとの比から音声区間を判別する．この手法は倍音構造の有無で判断しているため，突発的雑音のように倍音構造を持たない音に対してロバストとなる．

周期性成分と非周期性成分の比率は基本周波数幅の櫛型フィルタとスペクトルの内積により求められる．フレームごとに基本周波数を測定し，基本周波数幅の櫛型フィルタ (フィードフォワード型とフィードバック型の二種類) を作成する．二種類のフィルタは周波数応答の合計値が等しくなるように重み付けを行い，それぞれスペクトルとの内積を求める．この結果得られるフィードバック型との内積とフィードフォワード型との内積の比を求める．これによって，スペクトルのベクトルの向きが，フィードバック型とフィードフォワード型のどちらに近いのか計算する．

基本周波数とは，音声成分の中で最も低い周波数である．音声の母音等に見られる倍音構造は，この基本周波数を間隔とした構造となっている．上記の計算を行う上で，基本周波数は必要不可欠となる．クリーン音声の基本周波数を求めるのは簡単だが，雑音等が含まれた場合，最も低い周波数成分の周波数を算出する手法は用いることが出来ないため，精密に算出するのは難しい．

本稿ではこの基本周波数を櫛型フィルタにより求める．先にも述べたように，倍音構造は基本周波数を間隔としている．よって，櫛型フィルタと比較することで，基本周波数を算出することが出来る．スペクトルと 80Hz〜450Hz 幅の櫛型フィルタの内積を計算し，最も向きが近い櫛型フィルタの幅を測定する．なお，80Hz〜450Hz は音声の基本周波数があると思われる範囲であり、1Hz 刻みで櫛型フィルタを作成する．

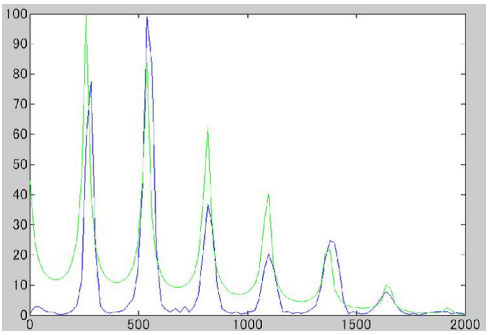

図 5. 櫛型フィルタを用いた基本周波数測定

4.2.3 二つの VAD の併用

参考文献 [7] では本稿と同じようにスイッチングカルマンフィルタ (SKF) による VAD と PARADE を併用して VAD システムを構築している．この文献では式 (1) のようにそれぞれに重み付けをし，和を求めて一定以上の値が求まったときに音声として認識している．なお g は GMM による VAD の結果，p は PARADE の結果，$\lambda(\tau)$ は固定値で，参考文献では $\lambda(\tau)$ に 0.4 を用いている．

$$Ps(\tau) = (1 - \lambda(\tau))g(\tau) + \lambda(\tau)p(\tau) \qquad (1)$$

この方法を試したところ，PARADE の結果は”S”等の無声音が倍音構造を持たないため，低い値になる．そのため，単語の開始位置と終了位置に関しては SKF による VAD の値で決定していることが多い．結果，この手法では PARADE は十分に SKF(もしくは GMM) による VAD の弱点を十分に補えていないことが分かった．

そこで本稿では，計算速度の速い PARADE を音声データに行い，PARADE で抽出した箇所に GMM による VAD を行う．PARADE の値が一定時間の間，閾値を超えていれば音声の可能性有りと判断し，GMM による VAD に音データを渡す．この際，無声音を見逃している可能性があるため，音声と認識した時間帯の前後 0.5 秒も含めて GMM による VAD に測定させる．この手法を用いることにより，参考文献の手法と比べて，誤検出率が低い．

4.2.4 VAD 評価実験

提案手法の VAD が本当に有効なのか調べる．そのために，VAD の適合率 (precision)，再現率 (recall)，及び VAD 使用後の音声認識の精度を，提案手法の VAD と julius の VAD それぞれ測定し比較する．適合率は抽出結果として得られた区間にどれだけ孤立単語が含まれているかという正解性の指標であり，再現率は抽出対象としている孤立単語の中で抽出結果とし

てどれだけ抽出することが出来ているかという網羅性の指標である．

GMM は特徴量として MFCC12 次元，ΔMFCC12 次元の合計 24 次元，混合数は 32 とした．なお，ケプストラム平均正規化法 (CMN) は使用しない．窓幅を 200ms，ずらし幅を 50ms で窓掛けして尤度計算を行う．音声は JNAS の音素バランス読み上げデータで学習した．発話者は男女合計 306 名で，一人につき約 50 文読み上げている．

PARADE は窓幅を 100ms，ずらし幅を 50ms で窓がけをして基本周波数測定，及び比率を算出した．

適合率は式 (2) によって，再現率は式 (3) によって求められる．なお，R は抽出された正解単語の数，N は抽出された単語の数，C は全サンプル中の正解単語数である．

$$precision = R/N \tag{2}$$

$$recall = R/C \tag{3}$$

従来手法として，julius の VAD を用いた場合と比較した．

検証に使用する音声サンプルは 20 単語 3 セット 10 人分の合計 600 単語を使用した．20 単語は Flash コンテンツ操作に最低限必要だと思われる命令である．録音環境は雑音の多いオフィスで行い，環境音として対話やファン音，キーボード音，咳，くしゃみ，音楽等が含まれている．なお，録音には単一指向性のヘッドフォンセットを用いた．

4.2.5 結果

提案手法の VAD は 600 単語中 574 単語認識し，その全てが正解単語だった．単語正解率は，julius の VAD 使用時の単語誤認率を 14% 改善出来る結果となった．VAD の検証結果を表 1 に示す．

表 1. VAD 評価結果

VAD	適合率	再現率	単語誤認識率
提案手法	100%	95.7%	14.0%
julius の手法	99.4%	99.8%	16.3%

600 サンプルでこの差に優位性があるのか調べる為，有意差検定を行った結果，従来手法と提案手法の差は誤差範囲内という帰無仮説は有意水準 1% で棄却された．よって，提案手法が単語正解率が従来手法よりも高いことが証明された．

適合率と単語正解率は従来手法よりも高い結果が得られたが，再現率は低い結果となった．提案手法で棄却された単語の平均 SN 比は 24.3dB となった．全体の平均 SN 比は 26.5dB で，棄却された単語との差はほとんどないといえる．また，単語ごとの棄却率に偏りが見られた．今回，検証を行った単語の中では，「スタート」や「スクロール」等母音を伸ばす箇所を含む単語の再現率が高かった．逆に，「オン」等のように単語自体が短いものや，「バック」等のように促音を含む単語の再現率が低かった．従って，再現率低下の原因は雑音の影響ではなく，単語ごとの PARADE の算出結果に偏りが発生していると考えられる．この理由として，促音を含む短い単語の場合、PARADE で高い値が得られるのは短い時間のため，音声だと判別出来なかったことが考えられる．音声信号の中でも，PARADE が高い値となるのは母音等の倍音構造が見られる箇所だけである．また，倍音構造を持たない雑音でもランダムに変動しているため，時折 PARADE の結果が高い値を示す．そのため，PARADE で短い区間に高い値が得られても，倍音構造があまり見られない音声なのか，雑音なのか判別出来ない．

5 提案手法による既存コンテンツの拡張

作成したプラグインを用いて実際にコンテンツを作成してみた．既存のコンテンツとして，3D オブジェクトを操作するものを使用する．これは，マウスやキーボードからのイベントを受け取ると，命令に応じて必要な処理を呼び出すように書かれている．

このコンテンツに音声認識制御関数を追加し，VAD と julius をインストールする．そして，必要な命令コードを音声認識制御関数に渡し，音声命令ごとに，どの関数を呼び出すのかを指定する．

実際にコンテンツを作動させた結果，3D オブジェクトを音声により操作することが出来た．しかし，1〜2 秒程度の遅延時間が生じた．

6 あとがき

単語認識の精度の向上，及び誤認を回避するために孤立単語認識を想定した VAD システムを作成した．

PARADE と GMM による VAD を用いた結果，適合率が十分満足出来る値となった．また単語誤認率は julius の VAD 使用時と比べて 14% 改善する結果となった．このことから，従来手法の VAD に比べて，提案手法の VAD を用いることにより，コンテンツを誤作動させにくいことが分かる．

しかし，本稿では遅延時間の短縮を行うことは出来なかった．人は対話インタフェースをマウス等手で操作する場合，遅延時間に 100ms 以上かかるとストレスを感じ始める [8]．この基準が音声による入力に当てはまるか検証する必要はあるが，仮に音声による入力でも 100ms 以上でストレスを感じると仮定する．実用性を考慮する上でも，音声による Flash コンテンツ操作も遅延時間を 100ms に抑えたいが，これは不可能だと考える．なぜなら，単語中に含まれる促音の無音区間と，単語と単語を区切る無音区間を区別をする必要がある．一般的に音声区間検出の測定基準として使用される NIST 基準では，音声信号区間は 300ms 以上の非音声区間で区切られると定義されている．つまり，300ms 以下まで遅延時間を短縮することは出来ない．しかし，逆に言うと 300ms までは遅延時間を短縮することが出来るといえる．よって，目標遅延時間を 300ms として短縮方法を研究したい．

また，単語ごとの再現率の値に偏りが見られた点については、どのような単語が認識しやすいか音素ごとに検証し，認識率の一覧を作成する必要がある．

参考文献

[1] Grace Chung, Stephanie Seneff, Chao Wang, Lee Hetherington, “A Dynamic Vocabulary Spoken Dialogue Interface”, *INTER-SPEECH*, pp. 327-330, 2005.

[2] 河原達也 他, “連続音声認識ソフトウェア Julius”, 人工知能学会誌, No. 20, pp. 41-49, 2005.

[3] Steven F. Boll, “Suppression of acoustic noise in speech using spectral subtraction”, *IEEE Trans.Acoust*, vol. ASSP-27, No. 2, pp. 113-120, 1979.

[4] 藤岡一馬 他, “ランニングスペクトルフィルタリングを用いた音声の雑音低減法”, 電子情報通信学会論文誌, vol. J88-D-II No. 4, pp. 695-703, 2005.

[5] Hyeopwoo Lee and Dongsuk Yook, “Space-Time Voice Activity”, *IEEE Transaction on Consumer Electronics*, vol.55, No.3, pp.1471-1476, October.2009.

[6] K. Ishizuka, T. Nakatani, M. Fujimoto, N. Miyazaki, “Noise robust front-end processing with voice activity detection based on periodic to aperiodic component ratio”, *Interspeech’07*, pp.230-233, 2007.

[7] S. Araki, M. Fujimoto, K. Ishizuka, H. Sawada, S. Makino, “A DOA based speaker diarization system for real meetings”, *HSCMA2008*, pp.29-32, 2008.

[8] S. K Card, T. P. Moran, A. Newell, “The psychology of human-computer interaction”, *Lawrence Erlbaum Associates*, 1983.