「アイサイ研究者のための音声情報処理ソフトウェア入門」特集号 解　説

隠れマルコフモデルに基づく日本語音声合成ソフトウェア入門

大浦圭一郎*・橋本　佳*・南角　吉彦*・徳田　恵一*

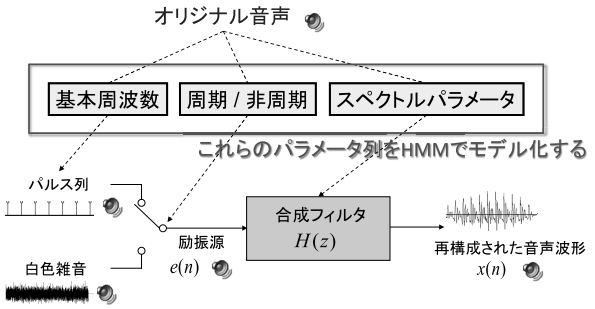
1. はじめに

音声合成技術とは，入力されたテキストに対し，明瞭で自然な音声を出力する技術である．古くから研究されてきており，近年ではカーナビゲーションシステム，電子書籍リーダ，音声対話システム，コミュニケーションロボット，音声翻訳システムなどで実際に使われるようになってきた．これまで,「フォルマント音声合成（1970年代）」,「ダイフォン音声合成（1980年代）」,「単位選択音声合成（1990年代）」[1],「統計的パラメトリック音声合成（1990年代後半）」[2] などのさまざまな音声合成手法が提案されてきており，ルールベースの手法からコーパスベースの手法が主流になりつつある．とくに統計的パラメトリック音声合成は，(1) 与えられた音声データに基づいてモデルを自動学習することにより，元話者の声質や発話スタイルを再現する合成音声を得ることができる，(2) 比較的少ない量の学習データで高品質な合成音声を得ることができる，(3) 学習用の音声データをランタイムのシステムに蓄積する必要がない，(4) モデルパラメータを適切に変換することにより，さまざまな声質や発話スタイルの合成音声を得ることができる，などの特徴があり，その中でも (4) はほかの手法では実現困難な特徴であり，声を真似る手法や混ぜる手法など，さまざまな手法が提案されている．本稿では，統計的パラメトリック音声合成手法の一つであり，近年の商用の音声合成製品にも採用されつつある「隠れマルコフモデル (Hidden Markov Model; HMM) に基づく音声合成手法」に注目し，その日本語実装であるオープンソースソフトウエア「Open JTalk」[3] を紹介する．以下，第**2**章ではHMM音声合成の概要について述べ，第**3**章では「Open JTalk」の導入方法を紹介する．第**4**章ではより実践的なカタマイズ法について紹介し，第**5**章でまとめる．

2. HMM音声合成の概要

2.1 音響パラメータ

音声の生成過程は，第1図に示すようなソース・フィルタモデルにより模擬することができる．音声波形から抽出された (1) スペクトルパラメータ，(2) 基本周波数，(3) 周期／非周期情報からなるパラメータの列を用いることで，元の自然音声を聴感上よく近似することができるため，これらの音響パラメータの列を入力テキストから推定することができれば，あらゆるテキストから音声を合成することが可能となる．



第 1 図　音声の分析と再構成

2.2 言語パラメータ

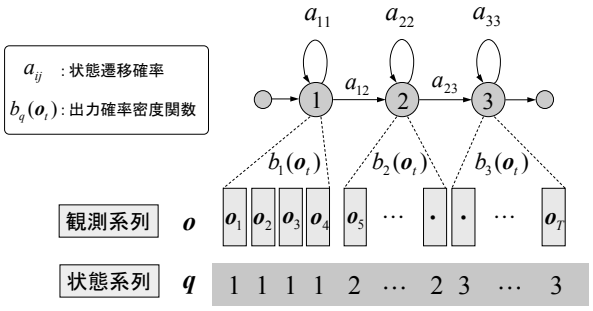
テキストはひらがな，カタカナ，漢字，数字などの文字の列によって構成される．各文字の発音は,「は」や「へ」が助詞のときに「わ」や「え」と発音する場合や,「110」の発音が「いちいちぜろ」「ひゃくじゅー」「ひゃくとーばん」になる場合など，その文字が属する単語の品詞や文脈などによってさまざまに変化するため，各文字と音響パラメータの対応関係をそのままモデル化することは難しい．そのため，品詞，未知語の発音，数字の読み，アクセント，母音の無声化・長音などを推定・抽出し，音素などの単位で連結した言語パラメータ列（ラベル）に変換してから用いる．

2.3 モデル化

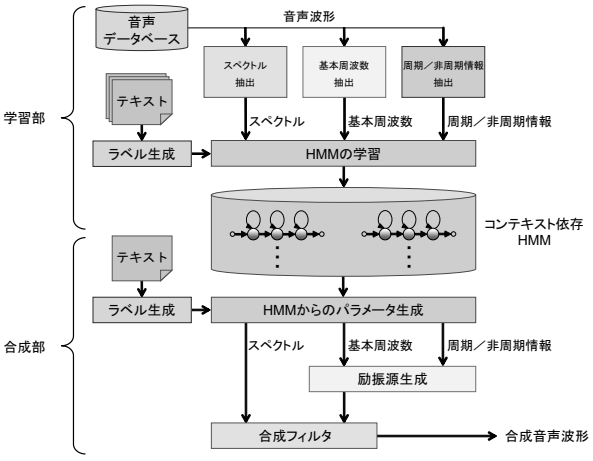
HMM音声合成では，音響パラメータ列とそれに対応する言語パラメータ列の対応関係をHMMによってモデル化する．第2図の観測系列が音響パラメータ列に対応しており，HMMは各状態における遷移確率と出力確率分布をモデルパラメータとして保持する．

HMM音声合成システムのブロック図は第3図に示すようなものとなり，図の上側が学習部，図の下側が合成部である．モデルパラメータである遷移確率と出力確率分布は，期待値最大化 (Expectation Maximization; EM) アルゴリズムによって推定することができる．HMMに

* 名古屋工業大学 大学院 工学研究科
Key Words: speech synthesis, hidden Markov model, text analysis, open source software.



第 2 図　隠れマルコフモデル



第 3 図　HMM 音声合成システム

よるモデル化の詳細については，文献 [2] を参照いただきたい．

3. 日本語音声合成ソフトウェア「Open JTalk」

本章では，HMM 音声合成手法の日本語実装であるオープンソースソフトウェア「Open JTalk」[3] について，インストールや使い方を紹介する．

3.1 インストール

「Open JTalk」は OS 依存性の低い実装であり，Windows，Mac OS，iOS，Android などでも動作を確認しているが，ここでは Linux (Ubuntu 16.04 LTS 64bit) 環境におけるインストール方法を紹介する．まず，最新のパッケージをダウンロード・解凍する（第 4 図の 1）．「hts_engine API」[4] は，HMM 音声合成の波形生成部分を実装した言語非依存のライブラリであり，日本語のテキスト解析が実装されている「Open JTalk」では，これを静的にリンクして使用する．本稿執筆時点の最新版はどちらもバージョン 1.10 である．

つぎに「hts_engine API」をコンパイル・インストールする（第 4 図の 2）．この際，“--prefix” オプションを用いて，インストール先（ここでは “/usr/local”）を指定する．インストールの完了後，インストール先の bin ディレクトリの中に，実行バイナリ “hts_engine” が生成されていることを確認する．“hts_engine” は，HMM 音声合成の波形生成部だけを動かすときに用いる実行バイナリである．

最後に「Open JTalk」をコンパイル・インストールする（第 4 図の 3）．この際，“--prefix” オプションを用いたインストール先（ここでは “/usr/local”）の指定は「hts_engine API」と同様だが，“--with-hts-engine-header-path” および “--with-hts-engine-library-path” オプションを用いて「hts_engine API」のインストール先を指定することに注意が必要である．インストールの完了後，インストール先の bin ディレクトリの中に実行バイナリ “open_jtalk” が生成されていることを確認する．

3.2 使い方

「Open JTalk」の動作を確認するため，音響モデルを用意する必要がある．ここでは「Open JTalk」のサイトから，男性 “m001” の音声を学習した音響モデルをダウンロード・解凍する（第 4 図の 4）．本稿執筆時点の最新版は，バージョン 1.05 である．

実際に音声を合成する際は，emacs や vi などのエディタで編集したテキストファイルを読み込む場合と，標準入力から読み込む場合の二通りの使い方がある．“open_jtalk” コマンドの必須オプションは “-x” オプションによる発音辞書ディレクトリの指定と，“-m” オプションによる音響モデルファイルの指定の二つである．実行する際のコマンドの例を第 4 図の 5, 6 に示す．なお，“-ow” オプションは，合成音声の波形ファイル名を指定するオプションである．読み込むテキストファイルや標準入力の文字エンコーディングは UTF-8（BOM 無し）である必要がある．

“open_jtalk” コマンドにはさまざまなオプションが用意されているが，ここではよく使うオプションを紹介する．

-x ディレクトリ名 発音辞書のディレクトリ名を指定するオプション．

-m ファイル名 音響モデルのファイル名を指定するオプション．

-ow ファイル名 合成音声の波形ファイル名を指定するオプション．

-ot ファイル名 合成時のログファイル名を指定するオプション．

-a 数値 声質を変更するオプション．デフォルト値は音響モデルファイルに設定されているため，ログファイルの “All-pass constant” を参照する．数値を減らすと声質が女性・子供に近づく．

-r 数値 話速を変更するオプション．デフォルト値は 1．数値を増やすと話速が速くなる．

-fm 数値 音高を変更するオプション．デフォルト値は 0．数値を増やすと音高が高くなる．

```
# 1. パッケージのダウンロードと解凍

% wget http://downloads.sourceforge.net/hts-engine/hts_engine_API-1.10.tar.gz
% wget http://downloads.sourceforge.net/open-jtalk/open_jtalk-1.10.tar.gz
% tar zxvf hts_engine_API-1.10.tar.gz
% tar zxvf open_jtalk-1.10.tar.gz

# 2. hts_engine APIのコンパイルとインストール

% cd hts_engine_API-1.10
% ./configure --prefix=/usr/local
% make
% sudo make install
% cd ..

# 3. Open JTalkのコンパイルとインストール

% cd open_jtalk-1.10
% ./configure --prefix=/usr/local --with-hts-engine-header-path=/usr/local/include --with-hts-engine-library-path=/usr/local/lib
% make
% sudo make install
% cd ..

# 4. 音響モデルのダウンロードと解凍

% wget http://downloads.sourceforge.net/open-jtalk/hts_voice_nitech_jp_atr503_m001-1.05.tar.gz
% tar zxvf hts_voice_nitech_jp_atr503_m001-1.05.tar.gz

# 5. 標準入力から音声を合成してWAVファイルに保存

% echo '今日の天気は晴れです。' | ¥
  /usr/local/bin/open_jtalk -x /usr/local/dic -m hts_voice_nitech_jp_atr503_m001-1.05/nitech_jp_atr503_m001.htsvoice -ow output.wav

# 6. テキストファイルから音声を合成してWAVファイルに保存

% echo '今日の天気は晴れです。' > input.txt
% /usr/local/bin/open_jtalk -x /usr/local/dic -m hts_voice_nitech_jp_atr503_m001-1.05/nitech_jp_atr503_m001.htsvoice -ow output.wav ¥
  input.txt
```

第 4 図　「Open JTalk」のダウンロードから実行までのコマンドの例

-g 数値 音量を変更するオプション．デフォルト値は 0．数値を増やすと音量が大きくなる．

-z 数値 合成音声をオーディオデバイスに送る際のバッファサイズを指定するオプション（**4.2** 節を参照）．デフォルト値は 0 に設定されており，0 の場合は合成音声をオーディオデバイスに送らない．

同じ文字列を入力した場合でも，“-a”, “-r”, “-fm”, “-g” を設定することでさまざまな合成音声を生成することが可能である．

4. より実践的な「Open JTalk」のカスタマイズ法

4.1 発音辞書の修正

「Open JTalk」は約 50 万語の発音辞書を搭載しているが，発音辞書に含まれない固有名詞や造語など，読み方が登録されていない単語に適切な読みを設定する場合は，発音辞書を修正する必要がある．ここでは地名「上終町」を「かみはてちょー」と発話させる場合の例を示す．まず，第 5 図の 1 の通り，修正前の挙動を確認すると，「上終町」を登録する前なので適切な読みになっていないことがわかる．

コンパイル前の単語エントリは “open_jtalk-1.10/mecab-naist-jdic/naist-jdic.csv” に記述されているため，このファイルに単語エントリを追加する必要がある．「Open JTalk」の発音辞書には「NAIST Japanese Dictionary」を使っており，解析時に品詞や活用情報が用いられるため，登録したい単語と同じ使われ方をする単語を見つけ，その単語を参考にしながら登録する形が簡単である．ここでは，地名「上終町」と似た使われ方をする単語エントリとして，同じ町名である「歌舞伎町」の単語エントリ “歌舞伎町,1353,1353,6883, 名詞, 固有名詞, 地域, 一般,*,*, 歌舞伎町, カブキチョウ, カブキチョー,0/5,C1” を参考にする（第 5 図の 2）．半角コンマで区切られた各カラムは，表記，単語 ID（左），単語 ID（右），単語コスト，品詞，品詞情報 1，品詞情報 2，品詞情報 3，活用型，活用形，基本形，読み，発音，アクセント型/モーラ数，アクセント結合規則となっている [1]．「歌舞伎町」を参考にすると追加すべき「上終町」の単語エントリは “上終町,1353,1353,6883, 名詞, 固有名詞, 地域, 一般,*,*, 上終町, カミハテチョウ, カミハテチョー,4/6,C1” の形になることがわかる．単語エントリの追加，発音辞書の再コンパイル・再インストール，動作確認をそれぞれ第 5 図の 3, 4, 5 に示す．もし追加した単語エントリが出現しない場合は，単語コストを下げることで対応可能である．

4.2 リアルタイム再生

音声対話システムなどで音声合成を扱う際，テキストを入力してすぐに合成音声の再生が開始される形が望ましい．ここでは，合成波形を一度ファイルに出力してから再生するのではなく，合成された波形を逐次的にオーディオデバイスに送信する方法を紹介する．

「Open JTalk」の波形生成モジュールである「hts_engine API」には，OS 依存性の高いオーディオインタフェースをラッピングしたライブラリである「PortAudio」を

[1] アクセントなどについては文献 [5] を参照．

```
# 1. 単語を追加する前の挙動の確認

% echo '上終町。' | ¥
  /usr/local/bin/open_jtalk -x /usr/local/dic -m hts_voice_nitech_jp_atr503_m001-1.05/nitech_jp_atr503_m001.htsvoice -ot output.log

% head output.log
  [Text analysis result]
  上,名詞,一般,*,*,*,*,上,ウエ,ウエ,5/2,C4,-1
  終,名詞,一般,*,*,*,*,終,オワリ,オワリ,0/3,C2,1
  町,名詞,接尾,地域,*,*,*,町,マチ,マチ,2/2,C3,1
  ...

# 2. 追加したい単語と似た使われ方の単語を検索

% grep ,歌舞伎町, open_jtalk-1.10/mecab-naist-jdic/naist-jdic.csv
  歌舞伎町,1353,1353,6883,名詞,固有名詞,地域,一般,*,*,歌舞伎町,カブキチョウ,カブキチョー,0/5,C1

# 3. 発音辞書に新しい単語を登録

% echo '上終町,1353,1353,6883,名詞,固有名詞,地域,一般,*,*,上終町,カミハテチョウ,カミハテチョー,4/6,C1' ¥
    >> open_jtalk-1.10/mecab-naist-jdic/naist-jdic.csv

# 4. 発音辞書の再コンパイルと再インストール

% cd open_jtalk-1.10
% make
% sudo make install
% cd ..

# 5. 単語を追加した後の挙動の確認

% echo '上終町。' | ¥
  /usr/local/bin/open_jtalk -x /usr/local/dic -m hts_voice_nitech_jp_atr503_m001-1.05/nitech_jp_atr503_m001.htsvoice -ot output.log

% head output.log
  [Text analysis result]
  上終町,名詞,固有名詞,地域,一般,*,*,上終町,カミハテチョウ,カミハテチョー,4/6,C1,-1
  ...
```

第 5 図 「Open JTalk」へ単語を登録する際のコマンドの例

静的にリンクして扱うことができる機能が装備されているため，まず,「PortAudio」をダウンロード・解凍する（第 6 図の 1）．本稿執筆時点の「PortAudio」の最新版は，バージョン 19.0600 である．

つぎに「PortAudio」をコンパイル・インストールする（第 6 図の 2）．この際，“--prefix” オプションを用いて，インストール先（ここでは “/usr/local”）を指定する．インストールの完了後，インストール先の lib ディレクトリの中に，静的ライブラリ “libportaudio.a” が生成されていることを確認する．

その後，「hts_engine API」を再度コンパイル・インストールする（6 図の 3）．この際，マクロ “AUDIO_PLAY_PORTAUDIO” を指定し，インストールした「PortAudio」へのリンクを LDFLAGS と LIBS を用いて指定することに注意が必要である.「Open JTalk」を再度コンパイル・インストールすることで（第 6 図の 4），インストール先の bin ディレクトリの中に生成された実行バイナリ “open_jtalk” が，リアルタイム再生に対応したものとなる．

オーディオデバイスへ合成波形を送る際のバッファサイズは “open_jtalk” コマンドの “-z” オプションで指定できる．小さい値を指定した場合は遅延を減らすことができ，一方，大きい値を指定した場合は音飛びに頑健な再生が期待できる（第 6 図の 5）．

4.3 オリジナル音響モデルの作成

「Open JTalk」にて話者を変更する場合，音響モデルファイルを入れ替える必要があるが，所望の声質や，感情を含むしゃべり方などを再現したい場合，発話データを用意することができれば，音響モデルを自作することが可能である．ここでは，ユーザ自身の音声をモデル化したオリジナル音響モデルの作成方法について紹介する．

まず，HMM 音声合成ツールキット「HTS」[6] とその学習スクリプトを用意する（第7図の 1, 2, 3）.「HTS」は HMM 音声認識ツールキット「HTK」のパッチとして公開されているため,「HTK」のサイトでユーザ登録をし，あらかじめ “HTK-3.4.1.tar.gz”, “HDecode-3.4.1.tar.gz” を入手しておく必要がある．また，学習スクリプトの実行には，音声信号処理ツールキット「SPTK」[7] が必要になるため，これもインストールする．

学習スクリプトには，あらかじめ言語パラメータ “HTS-demo_NIT-ATR503-M001/data/labels” と，男性話者 “m001” の音声波形 “HTS-demo_NIT-ATR503-M001/data/raw” が用意されている．言語パラメータと音声波形は同じ発話内容である必要があるため，ここでは発話内容を変えることなく，音声波形だけを入れ替えてオリジナル音響モデルの作成を実現する．発話内容は音声波形を wavesurfer などのツールで再生することで確認できるが，音声波形のファイルフォーマットはサンプリング周波数 48kHz，量子化ビット数 16bit, little-endian のヘッダ無しのファイルとなっているため，ファイルフォーマットを間違えないように注意が必要で

```
# 1. パッケージのダウンロードと解凍

% wget http://www.portaudio.com/archives/pa_stable_v190600_20161030.tgz
% tar zxvf pa_stable_v190600_20161030.tgz

# 2. PortAudioのコンパイルとインストール

% cd portaudio
% ./configure --prefix=/usr/local
% make
% sudo make install
% cd ..

# 3. hts_engine APIの再コンパイルと再インストール

% cd hts_engine_API-1.10
% ./configure --prefix=/usr/local CFLAGS='-g -O2 -D AUDIO_PLAY_PORTAUDIO -I /usr/local/include' ¥
  LDFLAGS='-L /usr/local/lib' LIBS='-lportaudio'
% make clean all
% sudo make install
% cd ..

# 4. Open JTalkの再コンパイルと再インストール

% cd open_jtalk-1.10
% ./configure --prefix=/usr/local --with-hts-engine-header-path=/usr/local/include --with-hts-engine-library-path=/usr/local/lib ¥
  LDFLAGS='-L /usr/local/lib' LIBS='-lportaudio'
% make clean all
% sudo make install
% cd ..

# 5. 標準入力から音声を合成して逐次的にオーディオデバイスに送信

echo '今日の天気は晴れです。' | ¥
/usr/local/bin/open_jtalk -x /usr/local/dic  -m hts_voice_nitech_jp_atr503_m001-1.05/nitech_jp_atr503_m001.htsvoice -z 3200
```

第 6 図　合成波形を逐次的にオーディオデバイスに送る際のコマンドの例

```
# 1. パッケージのダウンロードと解凍

% wget http://hts.sp.nitech.ac.jp/archives/2.3/HTS-2.3_for_HTK-3.4.1.tar.bz2
% wget http://hts.sp.nitech.ac.jp/archives/2.3/HTS-demo_NIT-ATR503-M001.tar.bz2
% wget http://downloads.sourceforge.net/sp-tk/SPTK-3.9.tar.gz
% tar jxvf HTS-2.3_for_HTK-3.4.1.tar.bz2
% tar jxvf HTS-demo_NIT-ATR503-M001.tar.bz2
% tar zxvf SPTK-3.9.tar.gz

# 2. HTSのコンパイルとインストール

% tar zxvf HTK-3.4.1.tar.gz
% tar zxvf HDecode-3.4.1.tar.gz
% cd htk
% patch -d . -p 1 < ../HTS-2.3_for_HTK-3.4.1.patch
% ./configure --prefix=/usr/local
% make
% sudo make install
% cd ..

# 3. SPTKのコンパイルとインストール

% cd SPTK-3.9
% ./configure --prefix=/usr/local
% make
% sudo make install
% cd ..

# 4. ユーザー自身の収録音声をコピー

% rm -f ./HTS-demo_NIT-ATR503-M001/data/raw/*.raw
% cp -f $directory_include_your_audio_files/*.raw ./HTS-demo_NIT-ATR503-M001/data/raw/

# 5. 学習スクリプトを実行

% cd ./HTS-demo_NIT-ATR503-M001/
% ./configure --with-sptk-search-path=/usr/local/bin ¥
      --with-hts-search-path=/usr/local/bin ¥
      --with-hts-engine-search-path=/usr/local/bin
% make
```

第 7 図　オリジナル音響モデルを作成する際のコマンドの例

ある．なお，学習スクリプトには 503 文の音声波形が同梱されており，収録文章数は多いほうが高品質になるが，合成音声品質を問わなければ 100 文程度でもよい．ユーザ自身の音声の収録後，その音声波形を元の音声波形と入れ替えて学習スクリプトを実行する（第 7 図の 4, 5）．マシンの性能にもよるが学習には半日ほどかかり，学習が終了すると，オリジナル音響モデルが “HTS-demo_NIT-ATR503-M001/voices” ディレクトリに生成される．作成したオリジナル音響モデルを「Open JTalk」で読み込んで，自身の声が再現されることを確認していただきたい．なお,「Open JTalk」は，音声インタラクションシステム構築ツールキット「MMDAgent」[8]

にプラグインの形で組み込まれているため，本節で述べた形で作成した音響モデルを「MMDAgent」の音声合成用の音響モデルとして読み込むことも可能である．

5. おわりに

本稿では，近年主流になりつつある統計的パラメトリック音声合成手法の一つである，隠れマルコフモデルに基づく音声合成の概要を紹介し，本手法の日本語実装であるオープンソースソフトウェア「Open JTalk」の導入方法を紹介した．また，単語エントリの追加方法や，合成波形を逐次的に再生するカスタマイズ方法，HMM 音声合成ツールキット「HTS」を用いたオリジナル音響モデルの作成方法について紹介した．HMM 音声合成ツールキット「HTS」は，統計的パラメトリック音声合成手法の中で近年盛んに研究が進んでいる「深層構造をもつニューラルネットワーク (Deep Neural Network; DNN) に基づく音声合成手法」にも対応しており，いずれ「Open JTalk」でも本手法が使えるように改良する予定である．音声合成関連の研究者でなくても，最新の音声合成手法が簡単に実現できるように，各種ソフトウェアの開発・公開を継続していきたい．

(2017 年 8 月 1 日受付)

参 考 文 献

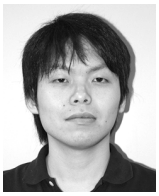
[1] A. J. Hunt and A. W. Black: Unit selection in a concatenative speech synthesis system using a large speech database; *ICASSP*, Vol. 1, pp. 373–376 (1996)
[2] H. Zen, K. Tokuda and A. W. Black: Statistical parametric speech synthesis; *Speech Communication*, Vol. 51, Issue 11, pp. 1039–1064 (2009)
[3] Open JTalk: http://open-jtalk.sourceforge.net/
[4] hts_engine API: http://hts-engine.sourceforge.net/
[5] 匂坂, 佐藤: 日本語単語連鎖のアクセント規則; 電子情報通信学会論文誌, J66–D.7, pp. 847–856 (1983)
[6] HTS: http://hts.sp.nitech.ac.jp/
[7] SPTK: http://sp-tk.sourceforge.net/
[8] MMDAgent: http://www.mmdagent.jp/

著 者 略 歴

大浦圭一郎

1982 年 2 月 28 日生．2010 年 3 月名古屋工業大学大学院工学研究科情報工学専攻博士後期課程修了．同年 4 月名古屋工業大学大学院工学研究科特任助教，2017 年 4 月名古屋工業大学大学院工学研究科特任准教授となり現在に至る．統計的パラメトリックな音声認識・音声合成に関する研究に従事．日本音響学会などの会員．

橋本 佳

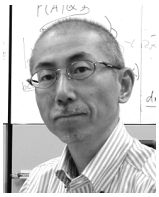


1984 年 3 月 1 日生．2011 年 3 月名古屋工業大学大学院工学研究科創成シミュレーション工学専攻博士後期課程修了．2012 月 4 月名古屋工業大学大学院工学研究科特任助教，2017 年 4 月名古屋工業大学大学院工学研究科特任准教授となり現在に至る．統計的パラメトリックな音声認識・音声合成に関する研究に従事．日本音響学会会員．

南角 吉彦

1977 年 3 月 10 日生．2004 年 3 月名古屋工業大学大学院工学研究科電気情報工学専攻博士後期課程修了．同年 4 月名古屋工業大学テクノイノベーションセンター大学院 VBL 部門中核的研究機関研究員，2005 年 4 月名古屋工業大学大学院工学研究科助手，2012 年 4 月名古屋工業大学大学院工学研究科准教授となり現在に至る．統計的パラメトリックな画像認識・音声認識・バイモーダル音声認識・音声合成に関する研究に従事．日本音響学会などの会員．

徳田 恵一



1960 年 12 月 9 日生．1989 年 3 月東京工業大学総合理工学研究科物理情報工学専攻博士課程修了．同年 4 月東京工業大学電気電子工学科助手，1996 年 4 月名古屋工業大学知能情報システム学科助教授，2004 年 4 月名古屋工業大学大学院工学研究科教授となり現在に至る．統計的パラメトリックな音声認識・音声合成に関する研究に従事．日本音響学会などの会員．