



ドリームキャストプラットフォーム用
ASR1600/C 音声認識アプリケーション開発ガイド

1. はじめに

人間の音声には個人差があります。たとえば、男性、女性、子供の音声は違いますし、言語によってアクセントが異なります。L&H ではこれらに対応できるように不特定話者音声認識技術を導入しているので、自分の音声の登録(録音)や認識率を高めるためのトレーニングをユーザー(話者)が行う必要はありません。これは、L&H ASR (Automatic Speech Recognition 以下 ASR) を使用すれば、連続音声を入力して使用する一般向け(不特定話者音声対応)アプリケーションの作成が可能であることを意味します。

L&H ASR 連続音声認識エンジンは言語非依存の音声レコグナイザであり、L&H でカバーしている言語ならば、すべて音声認識を行うことができます。音声認識エンジンには各言語について不特定話者モデルが用意されています。すべてのユーザーは、このモデルを利用できるほか、アプリケーションで定義したボキャブラリをこのモデルに登録することもできます。

ASR1600 音声認識エンジンは、L&H の連続音声認識ファミリー製品の最新エンジンであり、ASR1500 の機能を継承しています。

連続音声認識システムは、連続して発話された単語のフレーズを自動的に認識しますが、離散単語音声認識システムは、音声認識を行う際、単語間に短いポーズを入れる必要があります。L&H 音声認識エンジンは、普段の自然な話し方で認識できる連続音声認識機能を利用することができます。

L&H 音声認識エンジンは、連続音声認識以外にも、離散単語音声認識、キーワードスポッティング認識、連続数字音声認識なども利用することができます。これらの機能を使用するための特別なインターフェイスは必要ありません。

最新のリアルタイム連続音声認識システムでは、コンテキスト(例えばフレーズの予測情報など)を利用した音声認識処理を行うことができます。コンテキストはアプリケーションに依存しており、アプリケーション開発者が作成します。L&H ASR 開発ツールでは、アプリケーション開発者は L&H BNF グラマー プログラミング言語やグラマー コンパイラといったコンテキスト開発ツールを利用して、任意にコンテキストの作成を行うことができます。

以下の特殊用途のコンテキストが用意されています。

- 離散単語認識 : 離散単語認識可能な特殊なコンテキスト
- キーワード スポッティング認識 : フレーズ中のキーワードにスポットを当てる特殊なコンテキスト
- 連続数字文字列認識 : 数字文字列を認識する特殊なコンテキスト

L&H レキシコンツールキット (LexTool) では、各言語用に単語とその音声記号表記を収録した辞書を複数持つことができます。その 1 つである標準辞書は、各言語の単語をその音声記号表記に変換するためのエキスパートシステムとして装備されています。L&H では、各言語で使用される単語をすべてカバーした辞書を装備していますが、他の言語から派生している単語や、固有名詞、特殊分野の専門用語などについては、適当でない発音、あるいは全く誤った発音に変換されてしまうこともあります。L&H では、このような問題に対応できるよう、この種類の単語や音声記号表記を処理するインターフェイス機能を備えた例外辞書の機能を提供しています。

未エントリの単語をアプリケーション実行時にユーザーが単語リストに追加できる機能(ユーザーワード機能)も備わっています。ユーザーワード機能によって単語を追加する場合はタイプ入力は必要ありません。

2. コンセプトの定義

音声認識エンジン

実際の音声認識のタスクを実行する音声認識エンジンは、プラットフォーム非依存の API 1600/C¹ を通して操作します。

音声認識エンジン上では、すべてのデータはバッファとして受け渡しされるので、メモリへのロードはユーザー アプリケーションが行わなければなりません。音声認識エンジンが提供しているデータもありますが、開発ツールで作成することもできます。

言語

言語は、API 1600/C に組み込まれたコンセプトであり、各言語の音声認識に必要なすべての情報や知識を含んでいます。言語には以下のものが含まれます。

- 言語に依存する音声認識エンジンデータ
- 不特定話者ユーザー モデル データへのリンク

辞書

一般に辞書は、単語とその発音を記載したリストですが、音声認識システムの辞書は、グラマー コンパイラ、および単語追加機能を使用したコンテキスト作成の処理過程で使用されます。この場合、各言語ごとに辞書を指定できます。

また開発ツールからエキスパート システムへのアクセスにより、単語を音声記号表記に変換することもできます。このエキスパート システムは、該当する言語の辞書の役割を果たします。一方、例外辞書はエキスパート システムを拡張するものです。これは、エキスパート システムだけでは、他の言語から派生している単語、すなわち固有名詞、特殊分野の専門用語などが、適当でない発音、あるいは全く誤った発音に変換されてしまうことがあるためです。辞書は、特定の言語の単語をその音声記号表記に変換するエキスパート システムにアクセスするほか、例外発音も持つことができます。1 つの標準辞書が装備されています。

注：API1600/C から、これら辞書関連の機能を直接利用することはできません。

ユーザー

ユーザーは、ASRAPI² に組み込まれたコンセプトであり、音声認識エンジンにエントリされた話者 (ユーザー) に関連するすべてのデータやそのデータの管理方法など、音声認識のユーザー管理に必要な情報をすべて含みます。

不特定話者認識用の標準ユーザー データ (各言語対応) が、言語のインストール パッケージに収録されており、言語ごとにユーザーを追加登録することが可能です。

ユーザー コンセプトは、ASR データベースシステムの機能の 1 つです。ASR データベースは PC プラットフォームの開発環境には存在しますが、組み込みシステムにはありません。したがってユーザー コンセプトは API 1600/C には存在しないことになります。

¹ API 1600/C は ASR1600 コンシューマ API を表します。

² ASRAPI は、Win32 環境の ASR エンジンを表します。これは、L&H ASR 開発ツールで使います。

グラマー

連続音声認識では、ボキャブラリだけではなく、フレーズの単語間の関係を表すグラマーが必要です。グラマーとは、予測される音声入力がある一定の規則に基づいて記述する形式的記述です。使用頻度の高い簡単なグラマーは離散単語グラマーで、この場合、受け付けられるフレーズは特定のボキャブラリに含まれる単語になります。

一方、連続音声グラマーは、アプリケーション依存であり、特定のグラマー言語でアプリケーション開発者が記述します。グラマー コンパイラを使用してグラマーをコンパイルすると、コンテキストが生成されます。

L&H ASR 開発ツールには、BNF グラマー コンパイラが標準装備されており、L&H BNF グラマー言語で記述されたグラマーのコンパイルが可能です³。

構文

構文は、コンテキスト作成時にグラマー のコンパイルを必要としない、限定されたドメイングラマーとして定義されています。この構文を基準にコンテキストを作成することができます。構文には以下の種類があります。

- 離散単語構文 (IWS): 受け付けられるフレーズは、特定のボキャブラリに属する単語になります。
- キーワード スポットティング構文 (KWS): 受け付けられるフレーズは、特定のボキャブラリに属する単語を含むフレーズになります。
- 連続数字文字列 (CD) 構文: 受け付けられるフレーズは、任意の長さの数字文字列になります。

コンテキスト

開発ツールを使うと、グラマーをコンパイルして L&H 認識エンジンで使えるようにすることができます。このコンパイルされたグラマーをコンテキストと呼びます。コンテキストは、ユーザー、グラマー、シンボル、ボキャブラリなどの要素で構成されます。いずれも音声認識エンジンのコンテキストのコンセプトには必要不可欠なものです。

コンテキストは一連のフレーズを表すシンボルで構成されます。各コンテキストには、音声認識エンジン上でアクティブにできるエクスポートされたシンボルが、少なくとも 1 つ含まれています。

コンテキストは、クラス(単語クラス)を含むことができます。クラスとは、ある文脈において文法的に等価な単語のリストのことです。レキシコン ツールキット (LexTool) では、コンテキストを生成するグラマーのリコンパイルをすることなく、新しい単語をクラスに追加したり既存の単語をクラスから削除したりすることができます。クラスに属する単語の音声記号表記は後から変更することができます。この機能により、空のクラスを使用して、アプリケーション実行時にアプリケーション側から単語をコンテキストに追加したり、コンテキストから単語を削除するといった「動的コンテキスト」の用法が可能になります。

精度の高い音声認識結果が得られるように、グラマーやそのボキャブラリの作成には工夫が必要です。音声認識対象の単語セットを選択する際に以下に示すガイドラインに従うことにより、アプリケーションの音声認識の精度が大幅に向上することがあります。

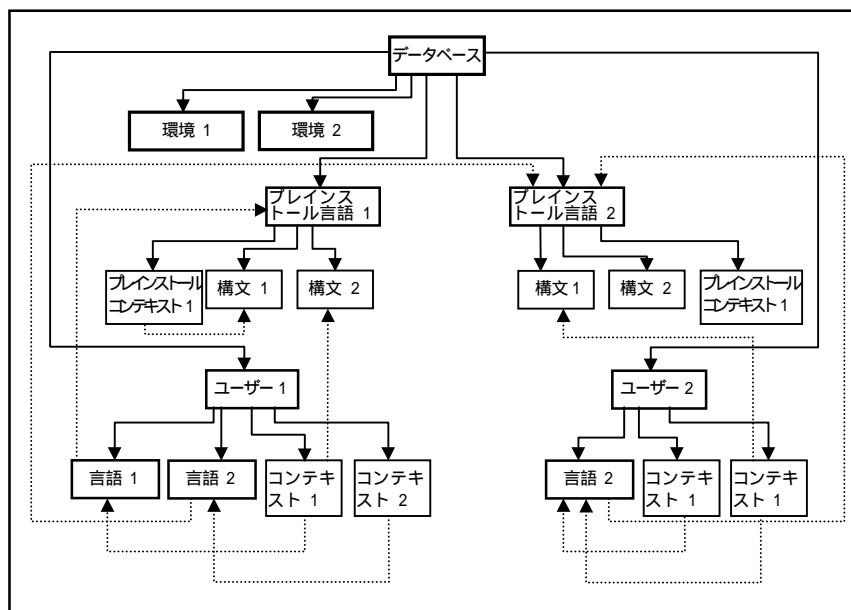
³グラマーの完全仕様については「Automatic Speech Recognition PC based Development Tools User Guide」を参照してください。

- 単語クラスの中で短い単語を使用することは極力避ける。単音節語は特に避ける。
- 単語クラスの中で発音の近い単語を使用することは極力避ける。
- コンテキストのパープレキシティ(1つの単語に続く単語の数の、コンテキストに含まれる単語全体についての平均値)を制限する。
- コンテキストの最大分岐数(1つの単語に続く単語の数の、コンテキストに含まれる単語全体についての最大値)を制限する。
- 1つの単語クラスに登録される単語の数を制限する。具体的には、現在の単語クラスをより小さな単語クラスに分割したり、すべての単語クラスが同時にアクティブになることを避けるように文法を修正したりする必要があります。たとえば、大量のデータが格納された従業員名簿は、所属部門ごとに分割し、従業員名の前に必ず所属部門を言わせるように文法を修正するとよいでしょう。

ASR データベース

ASR データベースとは、**PC 上の**音声認識エンジンに関連するすべての情報を集約的に格納したレポジトリです。L&H 開発ツールは PC 上の音声認識エンジン上に構築されています。認識エンジンに用いられるすべてのユーザー、言語、文法、およびボキャブラリがこのデータベースに含まれています。

ASR データベースは、複数のアプリケーションで同時に共有することが可能なため、新しいコンテキストのコンパイル、新しい言語のインストール、新規ユーザーの作成、複数のレコグナイザの起動などの操作が可能になります。音声認識アプリケーションの実行を妨げることはありません。ASR データベースの標準的な構造を以下の図に示します。



ASR データベースの構成要素は以下のとおりです。

- 環境
- プレインストール言語
- プレインストール構文
- プレインストール コンテキスト
- ユーザー
- 言語
- コンテキスト

環境

音声認識エンジンはさまざまなプラットフォームで実行することができます (Win16、Win32、RISC プロセッサ、および DSP ボード)。プラットフォームについての必要な情報は ASR データベースの「環境」に格納されています。

プレインストール言語

初期インストールではいくつかの言語がデータベースにインストールされます。プレインストール言語には不特定話者モデル、インストールされた言語の (標準的な) 例外辞書、新しい単語をコンテキストに追加する機能に必要な DLL が含まれます。

プレインストール言語は ASR データベース上のすべてのユーザー間で共有することができます。

プレインストール構文

各プレインストール言語ごとに、いくつかの構文もインストールされます。これは、プレインストール構文と呼ばれ、ユーザーがコンテキストを作成するのに使用します。ユーザーはその構文の単語クラスに単語を追加したり、削除したりすることができます。離散単語構文では、ユーザーは、その構文のただ 1 つの単語クラスに認識可能な単語を追加します。構文はそれぞれ、特定の言語にリンクしており、他の言語から使用することはできません。

構文は ASR データベース上のすべてのユーザー間で共有することができます。

プレインストール コンテキスト

各プレインストール言語ごとに、プレインストール コンテキストと呼ばれる定義済みのコンテキストがいくつかあります。コンテキストは、言語、構文、および単語の要素から構成されます。ユーザーは、自分専用のコンテキストも登録することができますが、このコンテキストにリンクしている言語を登録する必要があります。登録すれば、自分専用のコンテキストの単語クラスに単語を登録したり、単語クラスから単語を削除することができます。

コンテキストは ASR データベース上のすべてのユーザー間で共有することができます。

ユーザー

レコグナイザ上で ASR データベースを使用するには、まず、1 つ以上のユーザーの登録を行い、そのユーザーに 1 つ以上の言語を割り当てます。次に、プレインストール コンテキストを

登録するか、あるいはグラマーをコンパイルします。言語の登録により、その言語に関連するデータすべてがコピーされるので、ユーザー依存にすることができます。ユーザーがコンテキストを登録する場合も同様です。コンテキストを作成する場合、プレインストール構文を利用する方法とグラマー コンパイラを利用する方法の2通りが考えられます。前者の場合、ユーザーが追加登録した単語リストだけがそのユーザーに格納されます。後者の場合は、単語リストだけでなく、グラマー自体がそのユーザーにコピーされます。したがって、後者の場合、コンテキストの構文名は空欄になります。すなわちこのコンテキストはプレインストール構文とリンクしていないということです。

言語

各ユーザーは、自分専用の言語をいくつか登録することができますが、登録した言語以外のコンテキストの登録と作成を行うことはできません。

コンテキスト

ユーザーは、プレインストール コンテキストの登録、グラマー コンパイラを使用したコンテキストのインポート、プレインストール構文を使用した自分専用のコンテキストの作成が可能です。

3. アプリケーション コンテキスト開発

音声認識ユーザーインターフェイス作成に必要な開発サイクル(手順)を以下に示します。

対話式ユーザー インターフェイス (ダイアログ) 設計

これは、アプリケーションに必要な音声認識ユーザー インターフェイスをアプリケーション開発者が定義する概念設計の段階です。音声プロンプト、ボイス コマンド、関連するアクションをどのように定義するか、またダイアログの各状態における各コマンドの動作をどのように設計するかが鍵となります。

コンテキスト作成

概念設計とそこで定義された音声入力をもとに、その音声入力を形式的に記述するコンテキストを作成します。

L&H 開発ツールでは、コンテキスト作成用の開発ツールや、離散単語認識やキーワードスポッティング認識のためのボキャブラリの作成を支援する音声エキスパート システムが提供されているほか、連続音声コンテキストを作成するためのグラマー開発ツールも提供されています。

パフォーマンス テスト

L&H 開発ツールでは、作成したコンテキストの音声認識パフォーマンスをテストするためのエバリュエータ プログラムが提供されています。テストは、リアルタイムに行うことも、録音済みの音声ファイルを使用して行うこともできます。

4. コンテキストとドリームキャスト プラットフォーム用単語クラスバッファの作成

組み込みシステムで使用可能なコンテキストや単語クラス バッファを作成するには以下の手順にしたがってください。

1. BNF ファイルを作成して ASRAPI データベースにインポートするか、LexTool 開発ツールで構文に基づいたコンテキストを作成します。BNF ファイルの作成方法の詳細については、「Automatic Speech Recognition PC based Development Tools User Guide」を参照してください。
2. PC プラットフォーム上でコンテキストのパフォーマンス テストを行うことができます。
3. ASR 1600 エクスポート フォーマット を使用して LexTool 上のコンテキストをエクスポートします。その結果、コンテキストバッファと単語クラスバッファが書き出されます。
4. コンテキスト バッファは PC 上で使用することができます。単語クラス バッファは単語バッファとクラス バッファに分割する必要があります。分割するには、CONVWCL ツールを使用します。

LexTool により書き出されたバッファをドリームキャストのプラットフォーム上で使用するには、変換プログラム CONVWCL を使用します。このプログラムを実行するには、以下に示すコマンドを入力します。

CONVWCL 入力ファイル 出力ファイル 1^(*) 出力ファイル 2^(**)

(*) 出力ファイル 1 には出力するクラス バッファを指定します。クラス バッファのフォーマットについては「付録 A」を参照してください。

(**) 出力ファイル 2 には出力する単語バッファを指定します。単語バッファのフォーマットについては「付録 A」を参照してください。

5. CasrImportClass 関数を使用すれば、ドリームキャストのプラットフォーム上でクラス バッファを API に渡すことができます。ただし、単語バッファは API からは直接使用できません。この場合、アプリケーションで特殊な操作が必要になります。ctxfuncs.c ファイルには、アプリケーションが単語文字列のフィードバック情報を得るのに有効な関数が格納されています。ドリームキャスト プラットフォーム上で単語文字列の情報を使用したくない場合は、showctxinfo ツールを使用して、単語文字列から単語 ID を引き出すことができます。

付録 A

クラスバッファ

DWORD: このバッファのバイトサイズ
DWORD: クラス数 (Nc)
char sz[]: クラス名 0
char sz[]: クラス名 1
...
char sz[]: クラス名 Nc-1

単語バッファ

DWORD: このバッファのバイトサイズ
DWORD: 単語数 (Nw)
char sz[]: 単語名 0
char sz[]: 単語名 1
...
char sz[]: 単語名 Nw-1