Vol. 50 No. 4 情報処理学会論文誌 Apr. 2009

高度な実世界イベント認識を手軽に利用可能にする Instant Learning Sound Sensorの提案

根岸佑也[†1] 河口信夫[†1]

近年，加速度や音の信号解析による実世界コンテキストセンシング手法が多く提案されている．しかし，これらの信号解析手法を誰もが手軽に利用できるわけではなく，実世界イベント認識を用いたシステムの構築は容易ではない．本稿では，信号解析による実世界の音イベント認識を，部品のように手軽かつ低コストに利用可能にするスマートセンサと，対象音イベントに適した認識処理を本センサへ自動的に設定する Instant Learning 手法を提案する．本手法では，DP マッチングに基づく認識処理において，最も性能が良い特徴量の種類や窓長などの各パラメータの組み合わせを，認識率と誤認識率の評価の試行により自動的に選択する．これらにより，信号処理プログラミングを行うことなく，高度な実世界イベント認識が手軽に利用可能になる．実装したプロトタイプの評価により，様々な生活音などの音に対して，約 80%を超える認識率を持つ処理を自動設定可能なことを確認できた．また，イベント認識処理の計算量についても評価し，低コストなワンチップマイコン上にて動作可能であることを確認した．

Proposal of Instant Learning Sound Sensor for Easy Building of Real World Event Recognition System

Yuya Negishi[†1] and Nobuo Kawaguchi[†1]

Over the last few years, various context-sensing methods by signal-analysis of sounds and acceleration patterns in real-world events have been developed. However, it is not easy for everyone to utilize real world event recognitions by signal-analysis in developments of ubicomp applications. In this paper, we propose a smart sensor which can easily and less costly utilize a sound event recognition. We also propose an Instant Learning method which automatically configures an appropriate set of parameters in a DP matching based recognition process for the target event by evaluating several combinations of parameters. Based on our proposal, we designed and implemented an Instant Learning Sound Sensor. By evaluation experiment, we confirmed the smart sensor can automatically choose a proper set of parameters for various sound event with almost over 80% accuracy. Furthermore, we evaluated the required processing power and the memory consumption, and confirmed that the recognition process can be implemented on an one-chip microcontroller.

1. はじめに

我々の生活空間にセンサや計算機を埋め込み，生活や作業を支援，リッチなユーザ体験を提供するユビキタスコンピューティングにおいて，実世界から情報を取得することは重要な課題の一つである[6)10)21)]．これまで，実世界コンテキスト情報をセンシングするために，無線小型センサノードなどのデバイス，実世界情報抽出のための様々なデータ処理アルゴリズムが研究されてきた．センサデータの処理においては，温度・湿度・照度といったセンサデータを直接利用するだけではなく，複数のセンサデータを組み合わせて統合的に解析することや，時系列センサデータを信号解析することにより，取得可能な実世界情報を高度化する手法が提案されている．特に，マイクや加速度の時系列データに対する信号解析では，ユーザの行動や環境で起きたイベントといった実世界情報を詳細に抽出可能であり，多様な行動・状況を認識するシステムやアプリケーションが実現されている[1)2)3)4)15)17)]．

コンテキスト情報の種類ごとの抽出手法は，十分に多様化してきた．我々は種々の実世界コンテキストセンシング手法を，どのようにして幅広いタスクへ適用可能にするかの検討により，ユビキタスコンピューティングの発展に貢献できると考える．例えば，信号解析

†1 名古屋大学大学院 工学研究科 電子情報システム専攻
Department of Electrical Engineering and Computer Science, Graduate school of Engineering, Nagoya University

の専門家ではない人が，音や加速度の信号解析によるイベント情報の取得処理を自身のシステムに組み込む場合，信号処理プログラミングによるパターン認識処理の設計と実装を，簡単に実現可能とは言い難い．信号解析を伴う高度な実世界イベント情報のセンシングにおいても，手軽かつ低コストに実現できる部品デバイスが望まれる．すなわち，これまで提案されてきた多様なアプリケーションに適した実世界情報センシングのためのセンサ部品とデータ処理の中から，新たにシステムを構築するユーザが，欲しい処理を手軽に利用できる仕組みが必要である．その際に，信号処理に詳しくないユーザでも手軽に利用できるように，信号処理プログラミングが隠蔽されていることが望ましい．

本稿では，実世界コンテキスト情報を利用するシステムの構築を支援することを目的に，信号解析を伴う高度な実世界の音イベントセンシングを，部品のように手軽かつ低コストに実現するデバイスとしてのスマートセンサ (Instant Learning Sound Sensor) を提案する．さらに，対象音イベントに適した認識処理を本センサへ自動的に設定する Instant Learning 手法を提案する．本手法では，信号処理プログラミングをすることなく，信号解析による実世界イベント認識を手軽に利用可能にする．認識対象の環境音といったイベントの信号に対して，DP マッチングを基本とする認識処理における，そのパターンの認識に適した特徴量や窓長などの各パラメータの組み合わせを，認識率と誤認識率の評価を繰り返し試行しながら，最も性能が良かった処理やパラメータの組み合わせを選択する．また，多数のパターンを識別する音声認識や行動認識システムとは異なり，提案手法では，認識対象に特化することにより，認識処理を軽量化できる．これにより，実際に認識処理を実行するハードウェアの要件を下げ，ワンチップマイコンのような低コストで小型なデバイス上での実現を期待できる．

以降，2 節では，環境音や加速度を時系列データ解析することによる高度な実世界情報のセンシング手法について，関連研究を挙げながら説明する．3 節では，我々が提案する認識処理を選択するというメタな仕組みを持つ Instant Learning Sensor について基本的なコンセプトを述べる．4 節では，提案センサを実現するための実世界イベントの認識処理生成手法 Instant Learning について述べる．5 節では，提案手法を環境音認識に適用した Instant Learning Sound Sensor の設計を述べ，6 節にてプロトタイプの実装，7 節，8 節にて評価について述べる．最後に，9 節にてまとめを行う．

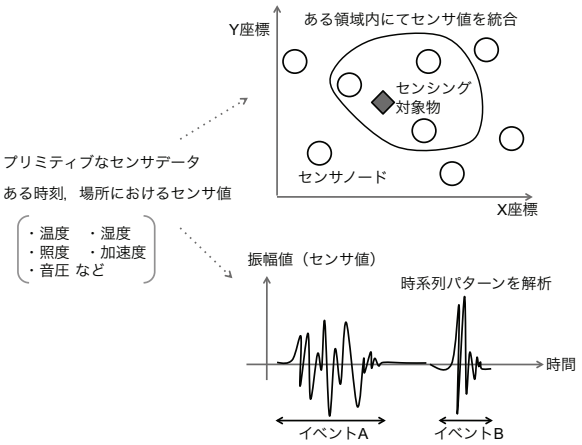

図 1 データ処理形式による分類
Fig. 1 Data Processing for Context Sensing

2. 実世界コンテキスト情報抽出のためのセンサデータ処理

環境側に埋め込まれた多数のセンサとユーザが身につけるセンサから，どのデータを選択し，どのようにデータを処理するのかは，ユーザが利用するサービスやアプリケーションごとに異なる．これまで，様々なユーザの状況や環境状態といったコンテキスト情報をセンシングするためのデータ処理手法やアルゴリズムが研究されてきた．センサデータの処理の形式は，次のように分類が可能である．(図 1)

- プリミティブなセンサデータの直接利用
  ある時点におけるセンサからの読み取り値を，そのまま利用する方法．例として，人感センサから取得した人の往来の有無情報，PDA のディスプレイ輝度を調整するために取得した照度情報[22]が挙げられる．
- 複数センサデータの統合
  センシング対象空間において，同種のセンサを複数個，もしくは，複数種類のセンサを配置し，ある時点における各センサ値を統合的に解析し，コンテキストを抽出する方法．例として，ある領域における平均温度を求めたり，複数の音響センサより得られる場所ごとの騒音レベルの分布から車の位置を推定する場合が挙げられる[12]．
- 時系列センサデータ解析
  一定期間に取得した時系列センサデータを信号処理を用いて解析し，コンテキストを抽出する方法．音声認識や計測工学などにおいて発展してきた信号解析的なアプローチを用いて，センサからの時

系列データを解析することにより，高度なコンテキスト情報を抽出可能である．例として，加速度データを用いて，ユーザのジェスチャーを時系列パターンマッチングにより識別し，行動情報を取得する場合が挙げられる[4)7)17)]．

プリミティブなセンサデータの直接利用は，読み取った情報をそのまま利用するのに対し，他の処理は 2 次的な情報への変換処理である．同一時刻における複数のセンサ値を統合するセンシング方法は，空間的にデータを切り出す処理と言える．時系列データを解釈するセンシング方法は，時間的にデータを切り出す処理と言える．データ統合や時系列解析によって，センサから取得可能な実世界コンテキスト情報を高度化させることできる．

2.1 時系列センサデータ解析と関連研究

前節にて挙げた時系列センサデータ解析を環境音や加速度へ適用し，詳細な実世界コンテキスト情報抽出を可能にするシステムや研究には，以下が例として挙げられる．

2.1.1 環境音の信号解析

Clarkson ら[4)]，Peltonen ら[17)] は，環境音認識を用いて，マイクから取得した環境音を解析し，ユーザの周囲の状況を取得する手法を提案している．Clarkson らの手法では，ユーザの肩に取り付けたマイクにて収録した環境音から，車の音，ドアを閉める音といった音イベント (Sound Object) をメル分割されたフィルタバンクのパワー，ピッチをといった特徴量の抽出により検出し，イベント区間をラベル付けする手法と，オフィス，スーパーマーケット，交通量の多い通りにいるといったシーン (Sound Scene) およびシーンの切り替わりを特定する手法を提案している．

Peltonen らは，同様に，車内，電車内，駅構内，通り，レストラン，図書館など屋内外を含む様々なシーンを音響情報のみによって識別する手法を提案している．識別処理の検討において，零点交差率 (Zero-crossing rate) などの時間的な特性や，パワースペクトルを特定の帯域幅ごとにまとめた Band-energy ratio などの周波数特性，メルケプストラム (MFCC) など，多様な特徴量を用い，シーン推定率を評価している．

環境音によって，ユーザを取り囲む環境状況だけではなく，ユーザの行動情報も詳細に識別可能である．アプリケーションも含めた例として，Chen ら[3)] は，バスルームにおいて手を洗う，シャワーを浴びる行動を，室内に取り付けたマイクを用いて認識し，健康管理を行うシステムを提案している．

2.1.2 加速度の信号解析

加速度センサは，ジェスチャー認識のような実世界指向ユーザインタフェースの研究[23)] などに応用されてきた．コンテキストアウェアネスへの応用としても，モノや服に加速度センサを取りつけ，詳細なユーザの行動コンテキストを抽出可能である．

コンテキストセンシングに応用する初期の研究[7)] では，走る，座る，寝ころぶなど基本的な動作の識別が行われてきた．Farringdon らが提案する Sensor Badge は，マイコンと安価な 1 軸加速度センサを二つ搭載した小型なウェアラブルデバイスとして設計され，身につけたユーザが立っている，座っている，寝ている，歩く，走るなどの状態を，加速度センサにかかる重力の方向と振幅の大きさから検出する．

より複雑なユーザのしぐさや行動を認識する手法として，Bao ら[1)]，Chang ら[2)] の手法が挙がる．Bao らは，ユーザの手首，腕，腰，膝，足首の 5 カ所に取り付けた 2 軸加速度センサを統合的に信号解析する手法を提案している．上述の基本動作に加えて歯を磨く，自転車を運転するなど 20 種類の日常行動を認識可能である．

3. 信号解析を伴う高度な実世界イベント取得の支援

2.1 節にて紹介した時系列データの信号解析手法により，ユーザの行動や環境状況で起きたイベントといった実世界情報を詳細に取得できることが示されている．しかし，信号解析の専門家ではない人が，そのような信号解析によるイベント情報の取得処理を自身のシステムに組み込む場合を考えると，信号処理を伴うパターン認識処理の設計と実装を，簡単に実現可能とは言い難い．我々は，信号解析を伴う高度な実世界イベント情報のセンシングにおいても，手軽かつ低コストに実現できる部品デバイスが望まれると考える．すなわち，これまで提案されてきた多様なサービス，アプリケーションに適したコンテキストセンシングのためのセンサ部品，信号処理といったデータ処理の中から，新たにシステムを構築するユーザが，欲しい処理を組み合わせ，手軽に利用できる仕組みが必要である．その際に，信号処理に詳しくないユーザでも簡単にイベント認識部品を構築できるように，信号処理プログラミングが隠蔽されていることが望ましい．

また，Huynh ら[11)]，Junker ら[13)] の研究では，取得したい実世界イベントに応じて特徴量を選択することにより認識性能を向上させることができること，低いサンプリング周波数やサンプリングビットの分解能

でも十分な認識性能を保つことができる可能性が示されている．これら対象イベントへの認識処理の最適化についても，上述のように手軽に利用できる仕組みが望ましい．

3.1 Instant Learning Sensor の提案

我々は，信号解析による詳細な実世界イベント認識を実現するための基本センサ部品として，次のような性質を満たすデバイス群があれば，信号処理プログラミングに慣れていないユーザでも，手軽に利用可能であると考えた．本稿では，コンセプトとして，時系列信号を伴う実世界イベントの認識処理を自動的に設定可能なスマートセンサ (Instant Learning Sensor) を提案する．

(1)**Instant Learning**：ユーザが検出したいイベントに対して，必要な信号処理（アルゴリズムやパラメータ）を自動的に選択可能

(2)**Smart Component**：部品デバイス単体でパターンマッチングを含めたイベント認識処理が可能で，他のデバイスやシステムと連携可能な部品であること

(3)**Simple Device**：低コストで小型なプロセッサなどのデバイスにて構成されること

提案システムは，スマート環境[24] やスマートオブジェクトなどの Do-It-Yourself やラピッド・プロトタイピングなど，広範囲に応用できる．具体例としては，多様なセンサやアクチュエータを搭載するブロック・デバイスをつなげることにより，容易にスマート環境を構築できる eBlocks[5] や Phidget[18]，Gainer[8] のようなシステムのイベント・トリガーとして，提案センサを組み合わせることが挙がる．

4. Instant Learning: 自動的な実世界イベント認識処理生成手法

本節では，Instant Learning Sensor のコンセプトを実現するための，自動的な実世界イベントの認識処理生成手法（Instant Learning）について述べる．

本手法では，入力された認識対象のパターンに対し，コンポーネントとしてあらかじめ保持しているフィルタ，特徴量，時系列パターンマッチング処理の組み合わせ，及び，各種コンポーネントのパラメータ調整を試行する．いくつかの組み合わせにて認識率と誤認識率といった性能を評価し，適切な性能が得られているか判断する．最終的に，試行した処理の中から最良の性能が得られた認識処理のコンフィギュレーションを出力する．例えば，環境音認識を対象とする場合，特徴量として，フレーム区間内の振幅値の平均・分散・零点交差数，パワースペクトル，ケプストラム，基本

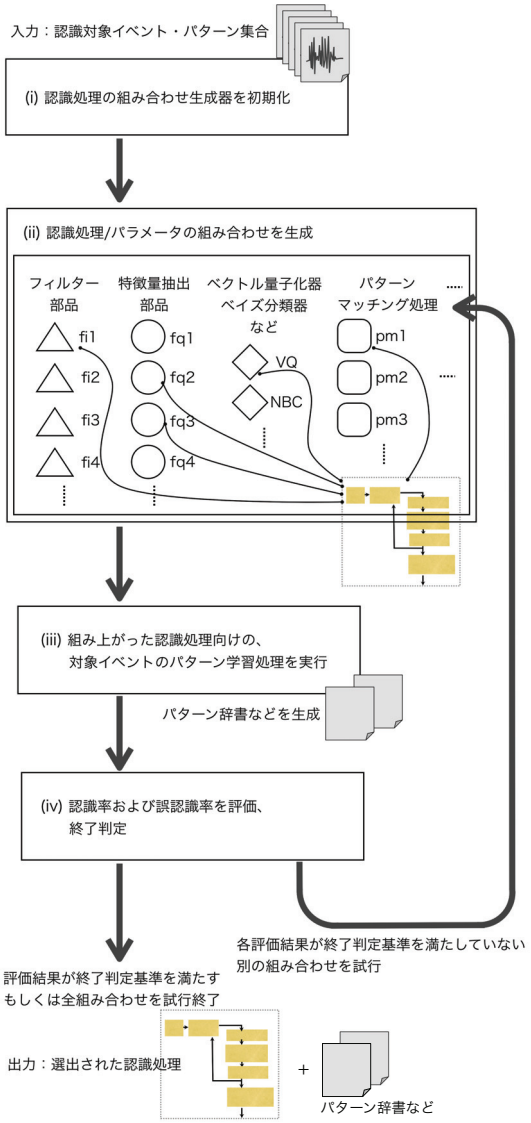

図 2 Instant Learning: アルゴリズム概要
Fig. 2 Process of Instant Learning

周波数の上昇と下降の度合いなどが挙げられる．パラメータとしては，特徴量ベクトルの次元数や，各特徴ごとの重みなどが挙げられる．時系列パターンマッチング処理としては，時間伸縮パターンマッチング（DP マッチング）[20] や隠れマルコフモデル（HMM）[19] が挙げられる．具体的な各種コンポーネントは，5 節にて，音認識に適用した場合について述べる．

多数の対象を識別可能な既存の音声認識や行動認識システムとは異なり，本手法では認識対象を絞ることにより，計算量とメモリ消費量を削減する．この結果，実際に処理を実行するセンサノードのハードウェア要件を下げ，低コスト化につながる．もし，複数のイベ

ントを認識させたい場合は，イベントの数だけ提案センサを用意する．

4.1 アルゴリズム

以下と図 2 に Instant Learning のアルゴリズムを示す．図中の番号は，以下の番号と対応する．

入力：ユーザにより与えられた認識対象/非認識対象のイベント・パターンの集合

(i) 認識処理の組み合わせ生成器を初期化：

各種認識処理用の信号処理コンポーネントの組み合わせを生成する探索アルゴリズムの初期値を決める．これは，用いるアルゴリズムに応じて行う．

(ii) 認識処理コンポーネントおよび各種パラメータの組み合わせを生成：

初期値もしくは，前に試行した組み合わせの評価結果を用いて，次に試行する認識処理を選択する．制約条件として，組み合わせ不可能なコンポーネントの除外，最終的に認識処理を実行するデバイスの計算速度，プログラム/データ・メモリ量が存在する．組み合わせ生成器は，単純な全探索だけでなく，膨大なコンポーネント群と広大なパラメータ調整範囲がある場合は遺伝的アルゴリズムなどの進化的計算，Simulated Annealing 法などの最適化アルゴリズムを適用する．

(iii) 認識対象のパターンを学習：

前項にて選択されたパターンマッチング処理に応じた認識対象パターンの学習アルゴリズムを実行する．例えば，DP マッチングを選択時には特徴量ベクトルを時系列順に格納したパターン辞書を生成し，HMM ならば状態遷移確率・出力確率を定める必要がある．また，パターンマッチング結果として得られる尤度や類似度よりイベント検出を判断する際の閾値も，ここで定める．他にも，ベクトル量子化を利用する場合は，コードブックを生成する必要がある．

(iv) 性能評価と終了判定：

入力された認識対象パターンおよび非認識対象パターンを用いて，試行されている認識処理の性能評価を行う．性能評価の指標としては，以下の式に示す非対象パターンの受理率 $P_{accepted}$ と対象パターンの棄却率 $P_{rejected}$，また，対象と非対象パターン集合におけるパターンマッチング結果の類似度や尤度の分布間距離が挙がる．類似度の分布間距離は遠いほど，対象と非対象のパターンを良く区別できる．評価の結果，探索を終了するか判定する．もし，規定の性能を得られた場合は終了し，得られなかった場合は再び (2) に戻り，別の組み合わせを試行する．

$$P_{accepted} = \frac{Accepted\ non\ targets}{Total\ of\ non\ targets} \qquad (1)$$

$$P_{rejected} = \frac{Rejected\ targets}{Total\ of\ targets} \qquad (2)$$

出力：選出された認識処理と学習結果データ

終了判定時に，全組み合わせを試行し終えた場合は，その中において最良の評価結果が得られた認識処理を出力とする．あわせて，選択された認識処理に関して，(iii) にて生成されたパターン辞書などのデータも出力する．

本手法を用いることにより，ユーザは信号処理プログラミングを行うことなく，信号解析を用いた実世界イベント情報抽出処理を作成できる．

5. Instant Learning Sound Sensor：音認識センサへの適用

4 節までにおいて，Instant Learning Sensor のコンセプト，および，認識処理の自動的な生成手法について述べた．我々は，本手法に基づき，生活音や環境音の認識に適用した Instant Learning Sound Sensor を設計した．

音は，豊富な情報を含むコンテクスト・メディアの一つである．生活音や環境音により，歩く，ドアを開け閉めする，掃除機をかける，テレビを見る，お茶を注ぐ，ティッシュペーパーを箱からを引き出すといった，多くの実世界イベントを認識できる．それらのイベントのいくつかは機械的なスイッチやモーションセンサのようなデバイスによって，手軽に認識可能である．しかしながら我々は，それぞれのイベントに対して，低コストかつ単一の音認識センサ・モジュールにて柔軟に対応できることに利点があると考える．以下，音イベント認識への適用について，設計と実装を述べる．

5.1 システム概要

Instant Learning Sound Sensor を次のように設計した．本システムを利用する上では，提案手法に基づき認識対象音を設定する段階と，実際にイベントを検出する通常動作の段階として，2 つの利用形態が考えられる．本稿では，各利用形態を次の 2 つのモードとして定める．

- **Event Learning Mode**：検出したい音イベント (Target Event Sound) を解析し，対象音を認識するのに適した認識処理を生成するモード．
- **Event Detection Mode**：実際に環境に配置されたセンサが，音イベントの有無を監視するモード．

実装プラットフォームとしては, マイコンのような安価なデバイスを想定している．しかしながら，Event Learning Mode は多くの認識処理の組み合わせを試

行するため，計算量が大きくなる．そのため，我々は，次のように，各モードを実行するデバイスを 2 つに分け，Sensor Configurator を計算機上で動作させることにした．

- **Sensor Configurator**：Event Learning Mode を処理するソフトウェアを動作させるデバイス．出力結果である音イベントの認識プログラムとコンフィギュレーションを ILSS-node に送る役割も担う．
- **ILSS-node**：Event Detection Mode を処理するデバイス．Sensor Configurator より生成された認識プログラムを実行する．音イベントが検出された場合，ネットワークなどを介して他のシステムに通知する．

また，音信号の取得のために，今回は圧電素子を振動センサとして利用する．圧電素子を選択した理由は，非常に低コストであることに加え，形状が小さく，家具や家電，室内設備，日用品など至る所に容易に貼付け可能なためである．また，一般的に，マイクを用いた環境センシングでは，周囲の会話音声など，他の音の影響を考慮する必要がある．振動センサの場合，素子が接していないモノを伝う振動の影響を受けにくいため，高度な雑音除去処理を省略可能であり，認識処理を軽量化できる利点がある．

5.2 音認識処理向け信号処理コンポーネントの設計

以下では，今回設計・実装した音イベント認識処理について述べる．組み合わせ生成器にて選択可能な項目を，周波数特性と振幅値特性に関する各特徴量の使用・不使用，特徴量の粒度などを示す特徴量ベクトルの次元数などの各種パラメータ値（表 1）とした．

5.2.1 特徴量抽出処理の検討

一般に音声認識においては，音声スペクトラムやメル-ケプストラム (Mell-Cepstrum)，振幅変化などを組み合わせ，音声の特徴量として利用することが多い．その他に，歌声や鼻歌認識による楽曲検索システム[25] においては，主に基本周波数の上昇，下降の変化が利用されている．既存の生活音，環境音による行動認識システムにおいては，音声認識と同様にメル-ケプストラムやパワースペクトラムなどが利用されている[3)15)]．

本システムが対象とする音の実例を図 3 に示す．各音の上部はスペクトログラムを表し，下部は振幅値の変化を示す．図より，それぞれの音に関して特徴があることが分かる．例えば，(a) 水道を流れる水の音は時間的変化を通じて周波数特性がほぼ一定である，(c) 机を 2 回ノックする音は，約 250msec 間隔で続く単発

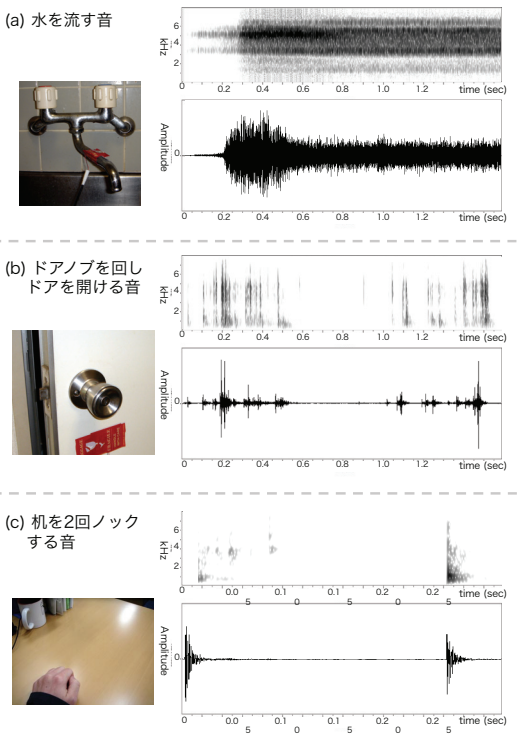

図 3 環境音および生活音の波形とスペクトログラムの例
Fig. 3 Example of Spectrogram and Waveform of environmental sound and life sound

音である．すなわち，認識対象となる音によって，選択可能な特徴量への重み付けを調整する必要がある．

5.2.2 パターンマッチング処理の検討

時系列に沿った特徴の変化が，認識対象の音信号と類似しているかどうかの判定処理には，音声認識において一般的に用いられる DP マッチング[20)]，および，HMM[19)] による尤度計算が挙げられる．HMM は，統計的に状態遷移モデルを構築するため，多量の学習データが必要である．本設計では，一つの ILSS-node にて多数のパターンを認識対象にしないことや，多量の学習データの準備に要するユーザの手間，処理の軽量さを考慮すると，DP マッチングが適切である．

5.2.3 基準パターンの保持

パターンマッチング時に参照する認識対象音を基準パターンと呼び，その特徴量ベクトルを，ベクトル量子化[9)] によって符号化し，代表ベクトルの集合であるコードブックの符号の系列として保存しておくことにより，メモリ消費量を削減できる．

5.2.4 音認識処理の設計

5.2.1〜5.2.3 節の検討をふまえ，低コストなマイコンでも動作可能な音認識処理の一例として，DP マッチングを主体に設計した．特徴量ベクトルとしては，

表 1 パラメータ一覧
Table 1 Parameter List

パラメータ	説明
$EnableMean$	振幅特性の要素として，窓区間内の振幅値の平均を利用・利用しない
$EnableVariance$	振幅特性の要素として，窓区間内の振幅値の分散を利用・利用しない
$EnableZerocrossrate$	振幅特性の要素として，窓区間内の零点交差数を利用・利用しない
$EnablePowerSpectrum$	周波数特性の要素として，パワースペクトルを利用・利用しない
F_{length}	窓長 (フレーム長)
F_{shift}	フレームのシフト長
W_f	特徴量ベクトル間距離における周波数特性要素に対する重み
W_a	特徴量ベクトル間距離における振幅要素に対する重み
N_v	周波数特性に関するベクトル次元数（パワースペクトル成分の分解能）
$CodebookSize\_S$	定常時音に関するコードブックに格納する代表ベクトル数
$CodebookSize\_T$	認識対象音に関するコードブックに格納する代表ベクトル数
$Threshold$	DP マッチングより得られた類似度に対する閾値

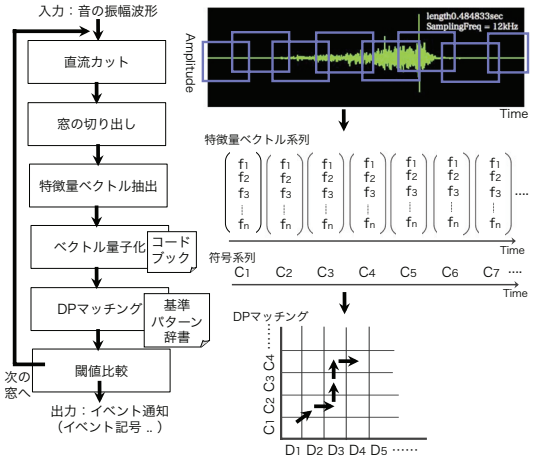

図 4 基本的な音認識処理
Fig. 4 Basic sound recognition process

短時間フーリエ変換 (STFT) より得られる周波数特性，および，振幅特性を要素として用いる．そして，特徴が時系列に沿って変化するパターンを，DP マッチングによって基準パターンとの類似度を計算し，閾値判定をする．認識処理の流れを図 4 に，調整可能なパラメータの一覧を表 1 に示し，以下に具体的な計算手順について述べる．

(i) ノイズ除去のために IIR フィルタを用いた直流カット処理を行う．

(ii) 入力された音の振幅データを窓区間に分割する．F_{length} 点が窓長，F_{shift} 点がシフト長である．

(iii) 窓区間ごとの特徴量ベクトルを求める．

(iii-1) 振幅特性に関して，平均，分散，零点交差数を求める．平均と分散は，窓区間内の各振幅値の絶対値に対して計算し，それぞれ，値が 0.0 ∼ 1.0 の範囲になるように正規化を行い，振幅特性に関する特徴量ベクトル V_a を得る．

(iii-2) 周波数特性に関して，Blackman 窓関数をかけた後，FFT 演算により $Flength/2$ 点のパワースペクトルを求める．得たパワースペクトルを均等に N_v 区間に分割後，0.0 ∼ 1.0 の範囲になるように正規化し，N_v 次元の周波数特性に関する特徴量ベクトル V_f を得る．

(iii-3) 特徴量ベクトル V_f を，後述する 5.3 節にて作成するコードブックと式 3 に示すベクトル間距離を用いて，ベクトル量子化する．特徴量ベクトルは，(平均, 分散, 零点交差数, 周波数特性をベクトル量子化後の代表ベクトル符号 c) を要素として持つ 4 次元の特徴量ベクトル V となる．

$$dis_f(v_{f,1}, v_{f,2}) = \frac{1}{N_v}\sum_{k=1}^{N_v}(v_{f,1}(k) - v_{f,2}(k))^2 \quad (3)$$

(iv) 得られた特徴量ベクトル V の時系列パターンと認識対象音の基準パターンの類似度を，式 3，4∼6 に示す各要素間の距離を用いて，DP マッチングにより求める．式中の W_f は周波数特性，W_a は振幅特性に関する重み，dis_c は代表ベクトル間の距離，dis_a は振幅特性の特徴量の距離，$Codebook(c)$ はコードブック中の代表ベクトル符号 c に対応するベクトルである．

$$dis(v_1, v_2) = W_f \times dis_c(c_1, c_2) + W_a \times dis_a(v_{a,1}, v_{a,2}) \quad (4)$$

$$dis_c\ (\ c_1, c_2) = dis_f(Codebook(c_1), Codebook(c_2)) \quad (5)$$

$$dis_a\ (\ v_{a,1}, v_{a,2}) = \frac{1}{3}((v_{a,1,mean} - v_{a,2,mean})^2 + (v_{a,1,variance} - v_{a,2,variance})^2 + (v_{a,1,zerocross} - v_{a,2,vzerocross})^2)) \quad (6)$$

(v) 得られた類似度（最小経路コスト）と閾値 $threshold$ を比較し，イベント検出を判定する．

一連の処理はフレーム・シフト長分の振幅データが

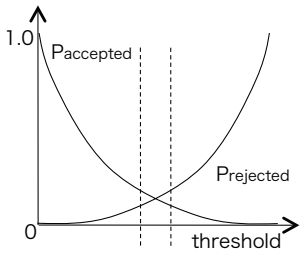

図 5 受理率・棄却率と閾値の関係
Fig. 5 Relation between threshold and $P_{accepted}$, $P_{rejected}$

入力される度に行い，リアルタイム認識を行う．また，一定時間，定常音が観測された場合，一つの音が入力し終えたものと見なす．

5.3 基準パターンの学習処理の設計

前節にて設計した音認識処理の実行時に必要なパターン辞書およびコードブックを生成する処理を述べる．この処理は 4.1 節の (iii) に相当する．

(i) ベクトル量子化時に用いるコードブックを作成する．組み合わせ生成器により選択されたパラメータ設定にしたがい，認識対象音より得られた周波数特性の特徴量ベクトル集合に対して，LBG アルゴリズム[14] を用いて代表ベクトルを選出し，コードブックとする．

(ii) 基準パターンの辞書を作成する．(i) にて得られたコードブックを用いて，周波数特性ベクトルの符号パターンおよび振幅特性のパターンを生成する．

(iii) 検出判定時の閾値を決定する．ユーザより与えられた認識対象音の集合と非対象音の集合を認識処理に通し，それぞれの集合ごとの DP マッチング結果である類似度の分布を得る．式 1，2 より，非対象音の受理率 $P_{accepted}$ と対象音の棄却率 $P_{rejected}$ を求め，$P_{accepted} = P_{rejected}$ となる点を閾値を最良とする（図 5）．

5.4 組み合わせ生成器

組み合わせ生成器として，表 3 に示す範囲のパラメータの全組み合わせを試行する．

表 2 に，図 3 の音に対する認識処理に適したパラメータ設定の例を示す．表 2 において，W_f，W_a が 0 であるもの，もしくは平均，分散，零点交差数のいずれかを利用しないものは，その特徴量に関する処理を行う必要がないことを意味する．他にも，計算量とメモリ消費量を削減し，小型・低コストなデバイス向けの軽量な認識処理を実現するために，可能な限り削減したパラメータ設定を行うことが望ましい．具体的には，特徴量ベクトルの次元数を削減することによりコードブックのサイズを減らすこと，窓長を長くしパターンマッチング時の基準パターンを短くすることが挙げられる．

表 2 パラメータ設定の例
Table 2 Example of parameter configurations

パラメータ名	音 (a)	音 (b)	音 (c)
F_{length}	256	512	512
F_{shift}	60	120	120
W_f	1.0	0.5	0.9
W_a	0	0.5	0.1
N_v	16	12	16
$Mean$	利用しない	利用	利用
$Variance$	利用しない	利用	利用
$Zerocrossrate$	利用しない	利用	利用
$PowerSpectrum$	利用	利用	利用
$CodebookSize\_S$	2	4	4
$CodebookSize\_T$	4	4	8

表 3 パラメータの探索範囲
Table 3 Range of parameters

パラメータ名	範囲
F_{length}	128, 256, 512
W_f, W_a	0.0 ～ 1.0 の範囲にて 0.1 刻み (ただし，$W_f + W_a = 1.0$)
N_v	12, 16, 32, 64
$Mean$	利用
$Variance$	利用
$Zerocrossrate$	利用
$PowerSpectrum$	利用
$CodebookSize\_S$	4
$CodebookSize\_T$	4, 8, 16, 32

5.5 終 了 条 件

組み合わせ生成器を全探索としたため，今回は特別な終了条件を設定しない．全組み合わせを試行後，式 2 に示す誤認識率の低さと，以下の式 7 に示す認識対象と非対象の類似度の分布間距離の遠さを用いて性能評価を行い，最良のパラメータの組み合わせを選出する．式 7 は，両分布間の距離の度合いを DP マッチングの結果の類似度 (*dpm costs*) の平均の比率によって求めている．r の値が高いほど，非認識対象音を入力した際に，誤認識しづらいことを示す．

$$r = \frac{Mean\ of\ dpm\ costs\ of\ non\ targets}{Mean\ of\ dpm\ costs\ of\ targets} \quad (7)$$

6. プロトタイプの実装

5 節の設計に基づき，Instant Learning の評価を行うため，計算機上に Instant Learning Sound Sensor のプロトタイプを実装した．

6.1 Sensor Configurator

実装した Sensor Configurator のスクリーンショットを図 6 に示す．Sensor Configurator は C++により，MacOS X 版と WindowsXP 版を実装した．

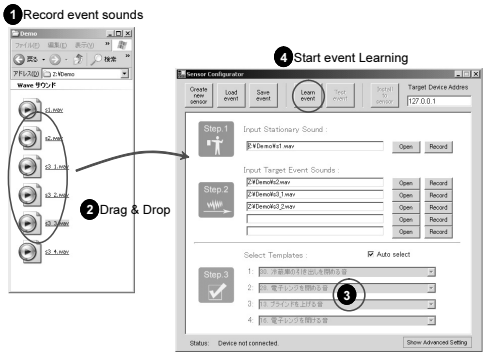

図 6 WinXP 版 Sensor Configurator のスクリーンショット
Fig. 6 Screenshot of Sensor Configurator on WinXP

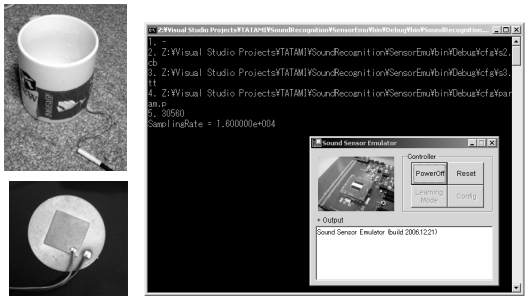

図 7 WinXP 版 ILSS-node ソフトウェアのスクリーンショット
Fig. 7 Screenshot of ILSS-node software on WinXP

図中の GUI には，定常音 Wav ファイル，認識対象の音イベント Wav ファイルを指定，もしくは録音するための機能を呼び出すボタンと，ユーザが用意した Wav ファイルを用いて Instant Learning を開始するためのボタンを備える．

6.1.1 ILSS-node ソフトウェア

生成された音イベントの認識処理を行う ILSS-node ソフトウェアを，C 言語により，MacOS X と Windows XP 向けに実装した．ILSS-node ソフトウェアは，Sensor Configurator より出力されたパラメータ設定，コードブック，パターン辞書を読み込み，PC 上のライン入力からの音，もしくは Wav ファイルより，認識対象音区間を検出する．イベント検出時には，コンソールにメッセージを表示することの他に，IP ネットワークを介し，イベント通知の UDP パケットを他のアプリケーションや端末に送信可能である．実際にコーヒーカップに圧電素子を貼付けた様子と，ソフトウェアの動作の様子を図 7 に示す．

7. Instant Learning の評価

前節で実装した Sensor Configurator と計算機上での ILSS-node ソフトウェアを用い，いくつかの音イベントについて認識率，誤認識率，および Instant Learning に要する時間を評価した．

7.1 評価用音イベントの収集

まず，評価に用いる音イベントとして，表 4 に示す音を含めた 30 種類の生活音・環境音を，それぞれ約 50 回分，合計約 1500 音を収集した．収録方法としては，図 3 のように圧電素子を日用品などに貼り付け，何もしていない定常時音と，実際にイベントを起こした時の音を繰り返し，計算機のマイク入力を通じてサンプリング周波数を 16kHz，サンプリングビットを 16bit にて録音した．

7.2 認識率の評価

収集した音イベントの中から 24 種類の音に対して，次のような評価を行った．

7.2.1 手　　順

(1) 認識処理の生成：それぞれの音イベント 5 回分を Sensor Configurator に入力し，音イベント認識処理を得た．

(2) 認識対象音の認識率：約 50 回分の対象音を与え，正しく検出された回数を測定し，認識率を算出した．$Recognition\ Rate(RR)$ は正しく検出された音の割合であり，$FalseRejectRate(FRR)$ は認識漏れの割合を示す．

$$RR = \frac{Accepted\ targets}{Total\ of\ targets} \tag{8}$$

$$FRR = \frac{Rejected\ targets}{Total\ of\ targets} \tag{9}$$

(3) 他イベント音入力時の誤認識率：認識対象の音イベントを除く他の 23 種類の音イベントを，不正解音として，それぞれ 5 回，合計 115 回分与え，誤検出された回数より，誤認識率 ($ER_{OtherEv}$) を算出した．

$$ER_{OtherEv} = \frac{Accepted\ non\ targets}{Total\ of\ non\ targets} \tag{10}$$

(4) 同一環境における非認識対象音の誤認識率：いくつかの音イベントの認識処理に対して，その音イベント収録箇所にて観測可能な非認識対象音を約 50 回分与え，(3) と同様に誤認識率 ($ER_{SameEnv}$) を算出した．

$$ER_{SameEnv} = \frac{Accepted\ non\ targets}{Total\ of\ non\ targets} \tag{11}$$

7.2.2 結　　果

結果を表 4 に示す．概ね 80% ～ 100% の認識率，0% ～ 20% の認識漏れ率にて，認識対象音を認識可能であることが確認できた．

いくつかの音イベントに関しては，70%未満の認識率および高い誤認識率となり，その音と他の音を識別

表 4 パラメータ全探索による認識処理の評価結果
Table 4 Result of evaluation of recognition processes

イベント名	認識率 RR	認識漏れ率 FRR	誤認識率 $ER_{OtherEv}$ (他のイベント音入力時)	同一環境における非認識対象音名	誤認識率 $ER_{SameEnv}$ (左記の音入力時)
1. 水道から水が流れる音	100%	0.0%	0.0%		
2. ドアノブを回し，ドアを開ける音	94.7%	5.3%	0.0%	ドアを閉める音	10.3%
3. 電話の受話器を下ろす音	100%	0.0%	14.2%	受話器を上げる音	7.0%
4. 引き戸を閉める音	98.2%	1.8%	0.0%	引き戸を開ける音	98.2%
5. コーヒーカップに水を注ぐ音	100%	0.0%	0.0%		
6. 引き出しを開ける音	79.8%	20.2%	0.0%	引き出しを閉める音	29.1%
7. AC タップにコンセントを差し込む音	86.5%	13.5%	1.7%	コンセントを抜く音	12.5%
8. 押しスイッチを押す音	96.4%	3.6%	0.0%		
9. ブラインドを巻く音	93.8%	6.2%	8.3%		
10. ガラス窓を開ける音	75.0%	25.0%	0.0.%	窓を閉める音	3.2%
11. 木製机を 2 回ノックする音	98.5%	2.5%	11.7%		
12. 電子レンジの扉を閉める音	91.0%	9.0%	0.0%	扉を開ける音	0.0%
13. プラスティック板を 2 回ノックする音	86.8%	13.2%	4.2%		
14. 冷蔵庫の扉を閉じる音	81.8%	18.2%	2.5%	扉を開ける音	0.0%
15. 冷蔵庫の扉を強く閉じる音	100%	0.0%	5.0%	扉を開ける音	0.0%
16. 冷蔵庫の扉を静かに閉じる音	92.8%	7.2%	2.5%	扉を開ける音	8.9%
17. 冷蔵庫の引き出しを閉める音	96.9%	3.1%	5.0%	引き出しを開ける音	2.3%
18. 静かに冷蔵庫の引き出しを閉める音	94.2%	5.8%	7.5%	引き出しを開ける音	24.6%
19. 金属板を 2 回ノックする音	80.4%	19.6%	0.0%		
20. 金属板を弱く 2 回ノックする音	98.6%	1.4%	8.3%		
21. ガラス窓の鍵を施錠する音	67.7%	32.3%	0.0%		
22. ホワイトボードにペンで書き込む音	100%	0.0%	94.2%		
23. ホワイトボードクリーナーで消す音	83.4%	16.6%	79.2%		
24. クリップボードにペンで書き込む音	68.7%	31.3%	20.8%		

表 5 処理時間計測結果
Table 5 Result of time of Instant Learning

イベント名	入力音の平均長 (msec)	全探索時 (sec)
1. 木製机を 2 回ノックする音	388	512
2. 電子レンジの扉を開ける音	1435	817
3. ドアノブを回しドアを開ける音	1529	978
4. ティッシュを箱から引き出す音	409	491

することが難しい場合もあった．これらは，その音が非常に短い音であるため，誤認識した音の一部分と高い類似性があったこと，もしくは，その音全体に関する周波数特性や振幅特性の変化が，誤認識した音と高い類似性があったことに起因する場合が多かった．その他に，ホワイトボードへのペンでの書き込みやクリーナーの音などは，圧電素子を通じて観測された音の振幅値が非常に小さかったため，認識率と誤認識率の両方が高くなり，単純な有音，無音の区別程度しかできなかった．これより，ユーザが本システムを用い，圧電素子をセンシング対象物に取り付ける際，音レベルが十分ではない場合は警告を表示するなどの措置が必要であると言える．

また，表中の誤認識率 $ER_{OtherEv}$ の結果は，異なる環境にて収録された音を使って実験を行っている点に注意されたい．すなわち，圧電素子を設置した対象物にて観測される音での誤認識率を示しているわけではない．

同一の環境にて収録された音を使った時の誤認識率は，誤認識率 $ER_{SameEnv}$ である．ドアノブを回しドアを開閉した時の音のように，反対の動作にて音の波形に顕著な違いが存在する場合は容易に誤認識を防ぐことが可能であった．一方，現在の特徴やパターンマッチング処理では識別不可能な例として，引き出しや引き戸をスライドして開け閉めする音が挙がる．両者に周波数成分の変化や，振幅値の変化にも大きな差がなく，識別が困難であった．このような音の認識を可能にするためには，新たな特徴量の導入が必要と考えられる．

7.3 Instant Learning 処理時間の測定

全探索を行う場合において，認識処理生成にかかる時間を測定した．実験環境として，MacOS X 版 Sensor Configurator を動作させる端末として，Apple MacBook (CPU: Intel Core Duo 2.0GHz, RAM:2.0GB,

OS: MacOS X 10.5) を用いた．

7.3.1 手　　順

前節の評価に用いた各音のいくつかに対して，結果を出力するまでに要する時間を計測した．表 3 に示すパラメータの範囲にて，528 通りの組み合わせを試行する．入力として，それぞれ 5 つの認識対象音の Wav ファイル，他の 23 種類の音イベントを非認識対象音として各 1 つずつの Wav ファイルを与えた．

7.3.2 結　　果

無音部分を除いた各 Wav ファイルの音の平均長 (msec) と，各処理の所要時間 (sec) を表 5 に示す．

本プロトタイプでは，設計に基づき，Instant Learning にかかる処理時間を削減するために，ILSS-node とは別の計算機上に Sensor Configurator を実装している．今回の結果は，比較的，狭いパラメータの探索範囲にて高速な CPU を持つ計算機上にて実行したため，約 5 分 ~16 分程度にて完了することができた．今後より多くの信号処理コンポーネントを追加した場合やパラメータの探索範囲を広めた場合を考慮し，効率的な最適化アルゴリズムを用いた組み合わせ生成器を実装することが望ましい．

8. ILSS-node における計算量の評価

本節では，ワンチップマイコンのような低コスト小型デバイスによる ILSS-node の実現可能性について述べる．6.1.1 節に実装した計算機向けの ILSS-node ソフトウェアにおける認識処理部分を，ワンチップマイコン向けに試作し，計算量を評価した．

8.1 小型デバイスにおける音イベント認識処理

今回，認識処理を動作させるマイコンとして，Microchip 社の dsPIC30F シリーズ (dsPIC30F6014A, RAM 8KB，ROM 144KB)[16] を評価対象に選択した．コンパイラには Microchip C30 (16 ビット C コンパイラ) を利用し，FFT や窓関数の乗算には付属の演算ライブラリを用いた．また，移植に際してマイコンに適した演算処理のために，可能な限り固定小数点演算を用いること，シフト演算にて除算を置換可能な箇所の変数の値域を 2 の累乗になるように正規化，コードブック内のベクトル間距離を定数テーブル化するなどの改良を行った．

低コストで小型な ILSS-node 向けのプロセッサに dsPIC30F シリーズを想定した理由には，(a) 低コストであること，(b) 消費電力が低く乾電池による駆動も可能であること，(c)FFT など信号処理向けの演算に適していること，(d) 数 MHz から最大 120MHz まで幅広い動作クロックを選択可能なこと，(e) 音データ入出力用の IC との接続インタフェース (Codec Interface) を備えていること，(f)ROM 領域が最大 144KB と比較的大きいことが挙げられる．理由 (e) の，接続可能な組み込み機器向けコーデック IC の例に，Siliccon Labratory 社の Si3000 Codec Chip (サンプリング周波数最大 12KHz，サンプリングビット 16bits) が挙がる．

表 6 (i) 直流カット，(ii) データ切り出しにかかる命令数，(v) 閾値判定
Table 6 Number of instructions for (i) and (ii) and (v)

	命令数
(i) 直流カット (IIR フィルタ)	493
(ii) 窓へのデータ切り出し	92 (窓長 $F_{length} = 512$, シフト長 $F_{shift} = 120$)
(v) 閾値判定	14

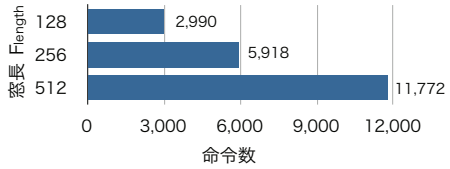

図 8 (iii-1) 振幅特性の特徴量の算出にかかる命令数
Fig. 8 (iii-1) Number of instructions for amplitude feature quantity

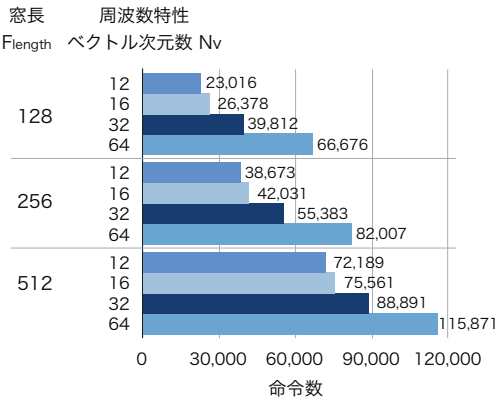

図 9 (iii-2) 周波数特性の特徴量の算出にかかる命令数
Fig. 9 (iii-2) Number of instructions for frequency feature quantity

8.2 命令数と実行時間

計算量の評価手順として，Microchip MPLAB IDE 上のシミュレータを用いて，表 3 に示した範囲で各パラメータを設定した場合の実行時命令数（オペコード数）を，5.2.4 節にて述べた処理 (i) から (v) 毎に測定

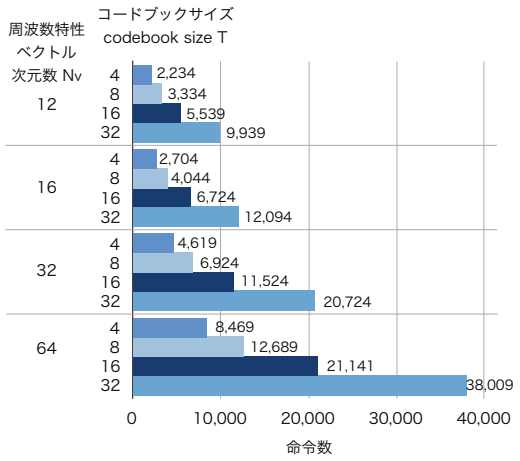

図 10 (iii-3) ベクトル量子化にかかる命令数
Fig. 10 (iii-3) Number of instructions for VQ

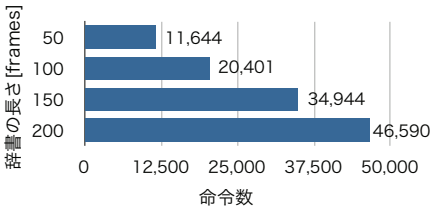

図 11 (iv)DP マッチングにかかる命令数
Fig. 11 (iii-3) Number of instructions for DP matching

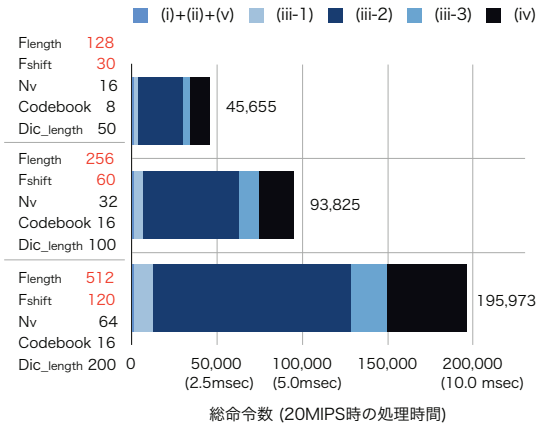

図 12 総命令数と実行時間 (20MIPS)
Fig. 12 Sum of instructions and processing time

表 7 メモリ使用量
Table 7 Memory consumption

	サイズ
ROM 領域 (プログラムおよび定数)	47742 bytes
RAM (グローバル変数)	6086 bytes
RAM (スタック)	1952 bytes
RAM 総使用量	8038 bytes

表 8 C 言語によるコード行数（認識処理のみ）
Table 8 Number of lines in recognition program

	行数
計算機上での認識処理	793 行
dsPIC30F6014A での認識処理	1617 行

した．結果を表 6，図 8 から図 11 に示す．各グラフの横軸は命令数を表す．実行時間は，(命令数)/(1 秒間に実行可能な命令数) [sec] にて求まる．dsPIC30F シリーズでは，どの命令も 4 クロックにて 1 命令を実行するため，実行時間は，4×(命令数)/(動作クロック MHz) $\times 10^{-3}$[msec] になる．

5.2.4 節で示す一連の周期処理は，窓のシフト長分の音の振幅値データを取得する度に行われ，次のシフト長分のデータを取得するまでに完了しなくてはならない．したがって，ILSS-node を実現するためには，(処理時間 [msec])≤(シフト長 [points]/ サンプリング周波数 [KHz]) が満たされる必要がある．例えば，サンプリング周波数 12KHz，30 点のシフト長（窓長 128 点）にした場合，2.5msec が処理時間の上限となる．シフト長が長い程，処理時間に余裕がでる．

図 12 にサンプリング周波数 12KHz において，シフト長を 30，60，120 とした場合に，制限時間内に実行可能なパラメータの組み合わせの例を提示する．20MIPS 動作（dsPIC30F シリーズでは 80MHz）では，それぞれ，2.5 msec，5.0 msec，10.0 msec 以内に周期処理の 1 サイクルが完了する．

8.3 メモリ使用量

表 7 に，メモリ使用量を示す．ROM 領域には，プログラム本体と，コードブック，コードブック中の代表ベクトル間の距離テーブル，基準パターンの辞書を保持する．表中のメモリ使用量は，窓長 F_{length} が 512 点，対象音のコードブックサイズ $CodebooksizeT$ が 32，周波数特性の特徴量ベクトルの次元数 N_v が 64，辞書の長さが 200 の時である．RAM 領域には，グローバル変数としてサンプリングデータ，FFT 演算，DP マッチングにおけるコスト計算に用いる配列などを保持している．また，一時的なローカル変数としてスタックを使用する．

表 7 より，プログラムおよび定数領域としては，50KB で十分なことが分かる．また，RAM は総使用量が 8KB で十分なことが分かる．今回，評価対象とした dsPIC30F6014A では，外部 ROM を利用することなく，要件を満たすことを確認できた．このことよ

り，計算量の面から，ILSS-node は低コストかつ小型なセンサノードとして十分に実現可能であると言える．

8.4 コード量

最後に，自動的にパラメータが選択される認識処理をユーザが利用可能になることにより，節約できるユーザの作業量の目安として，実装に要した認識処理のコード量（C 言語）を表 8 に示す．単純なコードの記述だけではなく，8.1 節にて述べたように，処理能力が低いワンチップマイコン向けの実装では，演算を高速に行うための最適化作業も必要である．提案手法によって，7.3 節の結果にて示したように，表 8 の実装にかかる作業を軽減することができる．信号処理の初心者に対しても，音イベント認識を利用したシステムの手軽な構築を支援可能であると言える．

9. ま と め

本稿では，信号解析を伴う高度な実世界イベントセンシングを手軽に利用可能にするための部品デバイスであるスマートセンサのコンセプトと，音イベントへの適用である Instant Learning Sound Sensor を提案した．そのために，認識対象に適した処理を自動的に選択するというメタな仕組みを持つ Instant Learning 手法を考案した．

音イベントに適用した本手法では，DP マッチングをベースとした認識処理において，ユーザが与えた認識させたい音イベントに適した特徴量や窓長などの各パラメータを，性能評価を繰り返し試行することにより，自動的に選択する．また，得られる認識処理は，ワンチップマイコンのような低コストな小型デバイス上での動作を想定している．そのために，提案手法では，ユーザが与えたイベントさえ認識可能であれば良いという考えに基づき，認識処理を軽量化する．

提案手法に基づき，プロトタイプを計算機上に実装し，自動的にパラメータが選択された処理の認識率を評価し，有用性を確認した．認識処理の計算量についても評価し，Instant Learning によって得られる処理が，小型デバイス上にて動作可能であることを確認した．

提案手法を用いることにより，信号処理に詳しくないユーザでも，信号処理プログラミングを行うことなく，手軽に自身のシステムへ音イベントセンシング機能を組み込むことができる．

今後の課題として，ILSS-node の実ハードウェア化，認識対象イベントの Instant Learning を行うためのユーザインタフェース，複数スマートセンサ間の連携などが挙げられる．

参 考 文 献

1) Bao, L. and Intille, S.S.: Activity Recognition from User-Annotated Acceleration Data, *In Proceedings of second International Conference on Pervasive Computing (Pervasive 2004)*, pp. 1–17 (2004).
2) Chang, K., Chen, M.Y. and Canny, J.: Tracking Free-Weight Exercises, *UbiComp 2007: Ubiquitous Computing*, pp.19–37 (2007).
3) Chen, J., Kam, A.H., Zhang, J., Liu, N. and Shue, L.: Bathroom Activity Monitoring Based on Sound, *In Proceedings of the 3rd International Conference on Pervasive Computing (Pervasive 2005)*, pp.47–61 (2005).
4) Clarkson, B., Sawhney, N. and Pentland, A.: Auditory Context Awareness via Wearable Computing, *In Workshop on Perceptual User Interfaces*, pp.37–42 (1998).
5) Cotterell, S., Mannion, R., Vahid, F. and Hsieh, H.: eBlocks - An Enabling Technology for Basic Sensor Based Systems, IPSN Track on Sensor Platform, *Tools and Desing Methods for Networked Embedded Systems (SPOTS)* (2005).
6) Dey, A.K. and Abowd, G.D.: Towards a Better Understanding of Context and Context-Awareness, *GVU Technical Report;GIT-GVU-99-22* (1999).
7) Farringdon, J., Moore, A. J., Tilbury, N., Church, J. and Biemond, P.D.: Wearable Sensor Badge & Sensor Jacket for Context Awareness, *In Proceedings of the Third International Symposium on Wearable Computers*, pp. 107–113 (1999).
8) Gainer: . http://gainer.cc/.
9) Gray, R.M. and Neuhoff, D.L.: Vector quantization, *IEEE ASSP Magazine, 1, 2*, pp.4–28 (1984).
10) Harter, A., Hopper, A., Steggles, P., Ward, A. and Webster, P.: The Anatomy of a Context-Aware Application, *Mobile Computing and Networking*, pp.59–68 (1999).
11) Huynh, T. and Schiele, B.: Analyzing features for activity recognition, *Proceedings of the 2005 joint conference on Smart objects and ambient intelligence*, pp.159–163 (2005).
12) Juan, L., James, R. and Feng, Z.: Collaborative In-Network Processing for Target Tracking, *EURASIP Journal on Applied Signal Processing*, Vol.4, pp.378–391 (2003).
13) Junker, H., Lukowicz, P. and Troster, G.: Sampling Frequency, Signal Resolution and the Accuracy of Wearable Context Recogni-

tion Systems, *Proceedings of the Eighth International Symposium on Wearable Computers (ISWC '04)*, pp.176–177 (2004).
14) Linde, Y., Buzo, A. and M.Gray, R.: An Algorithm for Vector Quantizer Design, *IEEE Transactions on Communications*, Vol. 28, No.1, pp.84–95 (1980).
15) Lukowicz, P., Ward, J.A., Junker, H., Stager, M., Troster, G., Atrash, A. and Starner, T.: Recognizing Workshop Activity Using Body Worn Microphones and Accelerometers, *In Proceedings of second International Conference on Pervasive Computing (Pervasive 2004)*, pp. 18–32 (2004).
16) Microchip: dsPIC30F6014A Data Sheet. http://ww1.microchip.com/downloads/en/DeviceDoc/70143D.pdf.
17) Peltonen, V., Tuomi, J., Klapuri, A., Huopaniemi, J. and Sorsa, T.: Computational Auditory Scene Recognition, *In Proceedings of IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP 2002)*, pp. 1941–1944 (2002).
18) Phidgets: Phidgets, INC. http://www.phidgets.com/.
19) Rabiner, L.R.: A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of the IEEE*, Vol.77, No.2, pp.257–286 (1989).
20) Sakoe, H. and Chiba, S.: A Dynamic Programming Algorithm Optimization for Spoken Word Recognition, *IEEE Trans. Acoustics, Speech, and Signal Processing,*, Vol.26, No.1, pp.43–49 (1978).
21) Schilit, B., Adams, N. and Want, R.: Context-Aware Computing Applications, *IEEE Workshop on Mobile Computing Systems and Applications*, pp.85–90 (1994).
22) Schmidt, A., Beigl, M. and Gellersen, H.-W.: There is more to context than location, *Computers and Graphics*, Vol.23, No.6, pp.893–901 (1999).
23) Wilson, A. and Shafer, S.: XWand: UI for Intelligent Spaces, *CHI 2003*, pp.545–552 (2003).
24) 中澤　仁，徳田英幸：スマート空間コンピューティング，日本ソフトウェア科学会, コンピュータソフトウエア，Vol.21, No.3, pp.55–65 (2004).
25) 園田智也，後藤真孝，村岡洋一：WWW 上での歌声による曲検索システム，電子情報通信学会論文誌, D-II，Vol.82, No.4, pp.721–731 (1999).