Instant Learning Sound Sensor: ユビキタス・コンピューティングのための柔軟なイベント音学習センサ

根岸 佑也† 河口 信夫†

本稿では，実世界における生活音や環境音を，小型・低コストなデバイスにより，コンテクスト情報として手軽に利用可能にすることを目的とし，認識対象音を即興的に学習可能な Instant Learning Sound Sensor を提案する．本システムでは，イベント学習時に，対象音ごとに適した特徴量抽出処理や認識アルゴリズムを自動的に選択し，認識処理を生成する．このため，ユーザは信号処理プログラミングを用いることなく，実世界の音を機器操作のイベント・トリガーのように利用するコンテクストアウェア・システムやユビキタス・サービスを手軽に構築することができる．実際に，設計とプロトタイプ実装を行い，小型・低コストなマイクロコントローラ上における実現可能性を確認した．また，評価実験により，様々な生活音に対して，約 83%を超える認識率を持つ生活音・環境音認識処理を自動生成可能なことを確認できた．

Instant Learning Sound Sensor: Flexible Sound Event Recognition System for Ubiquitous Computing

Yuya Negishi† and Nobuo Kawaguchi†

We propose a smart sound sensor component for building context-aware systems that can instantly learn and detect events from various sound information with a small and low cost device. Using the proposal sensor, a developer of a ubiquitous service can easily utilize a real world sound, like an event trigger to control appliances, without a signal processing programming. In the event learning phase, a developer is only required to input target event sounds from a microphone or sound files. The proposal system automatically analyzes and selects an appropriate sound recognition process. Actually, we designed and implemented the flexible event sound learning system on a PC, and sound sensor on a low cost Micro DSP device. By the experiment, we evaluate the feasibility of proposal sensor, and we confirmed it can automatically generate various event sounds recognition process with almost over 83% accuracy.

1. はじめに

近年，ユビキタス環境の実現の期待とともに，ユーザ自身の状況と取り巻くコンテクストに基づき，そのユーザに適したサービスを提供するコンテクストアウェア・システムが盛んに研究されている．コンテクスト情報を取得するデバイスとしては，GPS や加速度・圧力・温度センサなどが利用される．それらを集約した小型無線センサデバイス（MOTE[1]，Smart-Its[11]）も開発され，実際にスマートルームなどに組み込む実験もされている．Jianfeng Chen らは，バスルームにおける手を洗う，シャワーを浴びるといった行動イベントを，室内に設置したマイクを用いた音認識により取得し，健康管理に役立てるシステムを提案している[5]．Paul Lukowicz[9] らは，マイクや加速度センサを搭載するウェアラブルコンピュータを用いて，工房における行動認識システムを提案している．他にも，マイクや加速度センサからの時系列データを信号解析し，より詳細なコンテクスト情報を抽出するためのシステムや研究がいくつか提案されている[6)7)14)16)]．

しかしながら，そのような信号処理を伴う複雑なパターンの認識処理の設計は，個別の特徴量の解析などに手間がかかるため，日常空間中の任意の物音に対して，音認識処理を適用し，手軽にコンテクストアウェア・システムに活用するようなことは困難である．特にユビキタス環境においては，センシング対象のイベントは多種多様である．

本稿では，イベントとして検出したい時系列信号パ

† 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

ターンに対する小型・低コストなセンサ・デバイス向けの認識処理を，センサを設置する現場にて手軽に学習可能な Instant Learning Sensor を提案する．提案センサは次の３つの特徴を持つ．

(1)**Instant Learning**：センサ設置箇所にて，検出対象としたいイベントをユーザが実演することにより，提案センサに対象信号パターンを即興的に学習させることが可能．

(2)**Smart Sensor**：提案センサは認識処理を単体で処理可能であり，他の情報機器やデバイスと連携可能．実世界のイベント通知デバイスとして，コンテクストアウェア・システムの開発に組み込みことができる．

(3)**Simple Device**：本手法では，提案センサを日常空間への手軽な組み込みを実現するために，小型・低コストなマイクロコントローラ上での動作を目指し，可能な限りシンプルかつ軽量な認識処理の生成を行う．

Instant Learning を実現するために，本研究では，実演されたイベント信号を自動的に解析し，その信号パターンの認識に適した特徴量抽出処理や認識アルゴリズムを自動的に選択し，組み合わせ，結果の試行による認識率と誤認識率の評価を行い，そして，最も性能の良い認識処理を最適なものとしてユーザに提示する手法を考案した．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用いることなく，信号処理を用いた詳細なコンテクスト情報を取得できる．

Simple Device を実現するために，我々は１つのセンサは１つ，多くて２，３種類のイベントのみを認識可能であれば良いと考えた．認識対象を，少数の音に特化することにより，計算量とメモリ消費量を削減を可能にした．

上記の手法に基づき，最初の Instant Learning Sensor として，生活音や環境音の認識に特化した Instant Learning Sound Sensor を設計し，プロトタイプを PC と Microchip 社のマイクロ DSP コントローラである dsPIC[8] 上に実装した．マイクとしては圧電素子を用いた．

実際に，家具や日用品に貼付けた圧電マイクによって，コーヒーカップにコーヒーを注ぐ音や，ドアノブを回す音，水道から水が流れる音などを，80%を超える認識率が得られる処理を自動生成可能なことが確認できた．

本稿の構成は以下の通りである．まず第２節で，生活音や環境音を手軽にイベントとして認識可能なユビキタス・コンピューティング向け小型センサ・デバイスの有用性について議論する．続いて，第３節で軽量な環境音認識のための信号処理について検討し，提案センサに必要な課題を述べ，第４節で音イベントに関する即興的学習手法について述べる．第５節では第４節で提案した手法に基づく PC 上のプロトタイプ実装について述べ，第６節で評価を行い，第７節で小型デバイス化について述べる．最後に，第８節でまとめ，今後の課題を挙げる．

2. Instant Learning Sound Sensor とコンテクスト取得支援

ユビキタス・コンピューティングにおいて，実世界より情報を取得することは最も重要な課題の一つである．実世界と計算機上の世界を結びつけるために，現状の研究では，それぞれのタスクやアプリケーションに特化したデバイスやアルゴリズムに焦点が当てられることが多い．しかしながら，実世界のイベントの種類は非常に多様であり，それらのイベントに個別に対応していくことは，ほぼ不可能である．

本稿において，我々は小型・低コストなデバイス上にて動作する即興的学習機能を持つ音イベント認識センサを提案する．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．我々は，提案するスマート音センサのみを用いることにより，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と考えている．

当然ながら，それらのイベントの内，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他のデバイスによって，より手軽に認識可能なものもある．しかしながら我々は，それぞれのイベントごとに，そのようなデバイスを設計することは高価であり，煩雑であると考える．

一方，本システムは単一，かつ，低コストなデバイスという利点を持つ．加えて，非常に柔軟性があり，スマート環境の Do-It-Yourself やコンテクストアウェア・システムのラピッド・プロトタイピングなど，広範囲なアプリケーションに利用可能である．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[12] 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．また，家電操作だけでなく，携帯情報端末やスマートルーム，建物に組み込み，ユーザの行動をより詳細に取得可能なプレゼンス・サービスへの応用も考えられる．

そのようなセンサデバイスを実現するためには，低消費電力・低 CPU パワーなマイクロコントローラで

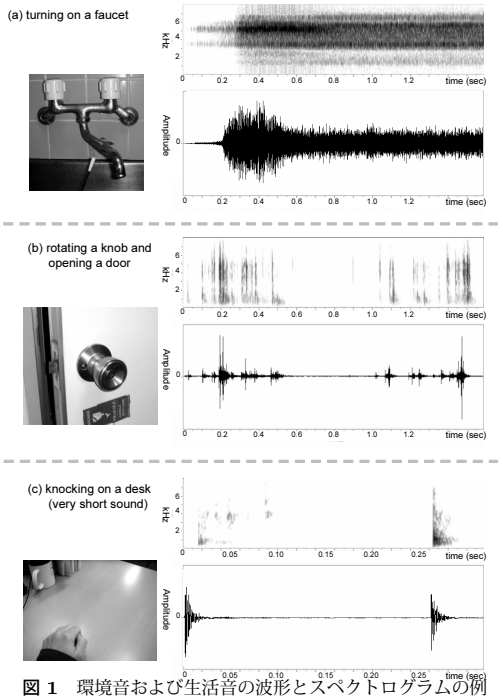

図 1 環境音および生活音の波形とスペクトログラムの例
Fig. 1 Example of Spectrogram and Waveform of environmental sound and life sound

も十分処理可能とするため，軽量な音認識アルゴリズムが必要である．また，使用可能なメモリ量も小さいため，認識の対象音を汎用化するのではなく，特定の音に限定することにより，認識に必要な配列データや定数データを削減の削減するアプローチが考えられる．

以上のことをふまえ，ユビキタス・コンピューティングに関連する分野に対する，本稿の主な貢献は以下の 2 点である．

(1) シンプルな 1，2 種類程度のみの音認識センサを提案．これにより，低コストなデバイス上での音認識を実現する．もし，複数のイベントを認識したい場合は，複数の提案センサを利用する．

(2) デバイス設置場所におけるセンシング・アルゴリズムとパラメータの設定手法の検討．これにより，単一のシンプルなデバイスに対して，柔軟な利用範囲を与える．

3. 生活音・環境音認識処理の検討

本節では，小型・低コストなマイクロコントローラ向けの生活音・環境音認識処理について述べる．

3.1 特徴量の検討

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，声の特徴量として利用することが多い[18]．

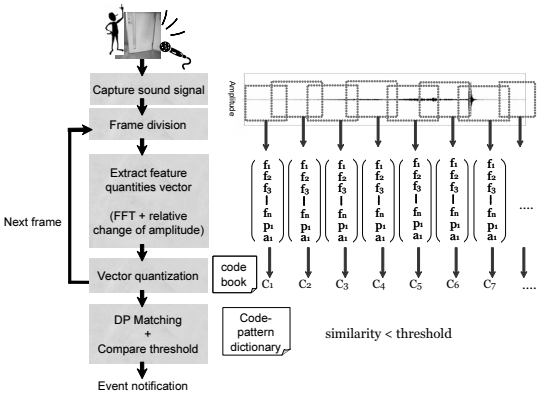

図 2 基本的な音認識処理
Fig. 2 Basic sound recognition process

その他に，歌声や鼻歌認識による楽曲検索システム[17]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[9)5)]．

本研究が対象とする音の実例を図 1 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，(b) ドアノブを回す音は，特徴的な振幅変化をしている．(c) 机を 2 回ノックする音は，約 250msec 間隔で続く単発音である．すなわち，認識対象となる音によって，適した特徴量の選択や，特徴量ごとに重み付けを行うことが必要である．

3.2 類似度計算の検討

時系列に沿って特徴が変化する信号が，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時間伸縮パターンマッチング (DP Matching, DTW)[2)4)]，および，隠れマルコフモデル (HMM) による尤度計算が挙げられる[18]．また，パターンマッチング時に参照する対象音を，そのまま保持しては必要なメモリ量が増大する．対象音の各時点における特徴量ベクトルをベクトル量子化[10] 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方が望ましい．

3.3 音認識処理の設計

前節における検討をふまえ，低コストなマイクロコントローラでも動作可能な音認識処理の一例として，処理の軽い DP マッチングを用いた処理を設計した．特徴量としては，1 回の Fast Fourier Transform(FFT)

表 1 パラメータ一覧
Table 1 Parameter List

パラメータ	説明
F_{length}	窓長 (フレーム長)
F_{shift}	フレームのシフト長
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
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_p	特徴量ベクトル間距離におけるパワー変化量要素に対する重み
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
L	式 1 における L
A_{max}	振幅要素の正規化時における振幅値の上限
$EnableLPF$	振幅成分に対するローパスフィルターの有効・無効
$CodebookSize\_S$	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
$CodebookSize\_T$	検出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

演算より得られる周波数特性，および，振幅特性を，対象音に応じて重み付けを調整する．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 2 に，パラメータ一覧を表 1 に示し，以下に具体的な計算手順について述べる．

(1) まず，入力された音の振幅データをフレームに分割する．この時，フレーム長を F_{length} 点，シフト長を F_{shift} 点とする．

(2) 続いて，各フレームごとに特徴量ベクトルを抽出する．ここでは Blackman 窓関数をかけた後，FFT を利用し，標本化定理より $Flength/2$ 点パワースペクトラムを算出する．得られたパワースペクトラムを均等に N_v 区間に分割し，各区間の平均値を求める．そして，各値を N_v 個の要素中の最大値で割り，0.0 ～ 1.0 の範囲になるように正規化を行う．これにより，N_v 次元の周波数特性に関する特徴量ベクトルを得る．

その他に，前フレームと現フレームの最大パワーを比較し，相対的なパワー変化量を $N_v + 1$ 番目の特徴量ベクトル要素として追加する (式 1)．式 1 中の $Pow_{max}(n)$ は，フレームの最大パワー，L は定数である．得られた値は，0.0 ～ 1.0 の範囲にて正規化する．

$$Vec_{N_v+1} = \frac{Pow_{max}(n) - Pow_{max}(n-1)}{L} \quad (1)$$

さらに，現フレームの振幅値の中から，最大振幅値の絶対値を $N_v + 2$ 番目の特徴量ベクトル要素として追加する．この時，振幅値に対して平滑化によってローパスフィルターを用いるかどうか，対象音によって選択可能にしている．そして，周波数特性と同様に 0.0 ～ 1.0 に正規化する．正規化する時には，振幅値の上限を A_{max} とする．

最終的に $N_v + 2$ 次元の特徴量ベクトルを得る．

(3) 得られた特徴量ベクトルをベクトル量子化する．この時に使用するコードブックは，後述するイベント学習処理において作成され，定常音コードブックと対象イベント音コードブックの 2 種類のコードブックを用いる．より入力されたベクトルとコードブック中の代表ベクトル間の距離が近い方のコードブック中の代表ベクトルの符号に変換する．ベクトル間の距離 $dis(v_1, v_2)$ の計算式を式 2 から 5 に示す．この時，対象音によって周波数特性の変化が顕著なもの，振幅変化に特徴があるものなどがあると考えられるため，(2) にて計算した特徴量ベクトル中の周波数特性，パワー変化量，振幅値の 3 つの成分に対して重み付けを行う．式中の W_f は周波数特性，W_p はパワー変化量，W_a は振幅に関する重みである．

$$\begin{aligned} dis(v_1, v_2) &= W_f \times dist_f(v_1, v_2) \\ &+ W_p \times dist_p(v_1, v_2) \\ &+ W_a \times dis_a(v_1, v_2) \end{aligned} \quad (2)$$

$$dis_f(v_1, v_2) = \sum_{k=1}^{Nv} (v_1(k) - v_2(k))^2 \quad (3)$$

$$dis_p(v_1, v_2) = (v_1(N_{v+1}) - v_2(N_{v+1}))^2 \quad (4)$$

$$dis_a(v_1, v_2) = (v_1(N_{v+2}) - v_2(N_{v+2}))^2 \quad (5)$$

(4)(3) より得られた各フレームごとの符号を，時系列に沿って保持しておき，検出対象のイベント音の符号系列と DP マッチングをし，類似度（最小経路コスト）を求める．そして，得られた類似度を表 1 の $threshold$ と比較し，検出対象の音が入力されたか閾値判定する．

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．

3.4 パラメータ設定と課題

図 1 の音に対する認識処理に適したパラメータ設定

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

parameter name	sound (a)	sound (b)	sound (c)
F_{length}	256	128	512
F_{shift}	60	30	120
N_v	16	12	32
W_f	1.0	0.4	0.5
W_p	0	0	0.5
W_a	0	0.6	0
L	-	-	120.0
A_{max}	-	30000	-
$EnableLPF$	false	false	false
$CodebookSize\_S$	2	4	4
$CodebookSize\_T$	4	16	16

の例を表２に示す．これは，人手による試行錯誤の結果，経験的に決定されたものである．表２において，W_f，W_p，W_a が０であるものは，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は，特徴量ベクトルの次元数やコードブックのサイズを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．

また，本研究では，既存の環境音認識と異なり，未知のイベント音を認識するための処理を自動的に生成することを目的としている．そのため，対象とする音に応じて，どのような特徴量や認識アルゴリズムが適しているのか，どのようなパラメータが最適であるか探索を行うことが必要である．

本稿では，あらかじめ多様な生活音・環境に対応する認識処理を備えておくことにより，対象のイベント音によって適応的に認識処理やパラメータのセットを選択可能な柔軟なコンフィギュレーション手法を提案する．それにより，未知のイベント音への対応を実現する．次節では，Instant Learning Sound Sensor のシステム設計，具体的な生活音・環境音認識処理の設計と柔軟なコンフィギュレーション手法について述べる．

4. Instant Learning Sound Sensor の設計

4.1 システム概要

Instant Learning Sound Sensor を次のように設計した．提案システムは次の２つのモードを持つ．

- **Event Learning Mode**：検出したいイベント音 (Target Event Sound) を解析し，対象音を認識するのに最適な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，イベント音の有無を監視するモード．

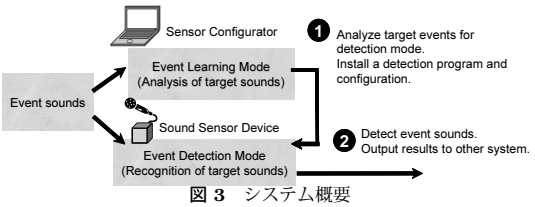

図 3　システム概要
Fig. 3　System Overview

Event Learning Mode では，対象音は実際に開発者がイベントを実演することによって入力を行う．それにより，手軽な学習操作を実現する．そして，入力された対象音を基に，自動的に幾つかの認識アルゴリズムおよびパラメータの組み合わせを試行し，認識率・誤認識率を評価する．最終的に最も良い評価が得られた認識処理を Event Detection Mode に引き渡す．

また，提案システムは２つのコンポーネントより構成される．

- **Sensor Configurator**：Event Learning Mode を処理するソフトウェア．出力結果である対象イベント音の認識プログラムとコンフィギュレーションを Sound Sensor Device に書き込む役割も担う．
- **Sound Sensor Device**：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．イベント音が検出された場合，ネットワークを介して他のシステムに通知する．

各モードとコンポーネントの関係を図 3 に示す．

実装プラットフォームとして，Sensor Configurator は PC のように計算能力の高い端末を想定する．一方，SoundSensor Device の方は，2 節および前節で述べたように小型・低コストなマイクロコントローラ，もしくは携帯可能な端末上への実装を想定している．

また，マイクとして，今回は圧電素子を利用する．圧電素子を選択し理由は，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点に着目したためである．また，一般的に，マイクを用いた環境センシングでは，環境内の雑音によって認識率が著しく低下する傾向にあるが，圧電素子は通常のマイクと比較して，素子が接している物の振動や近接する音しか取得しないため，環境ノイズ，人間の会話音声などの雑音の影響を受けにくい．そのため，雑音除去処理を省略可能ということは，認識処理を軽量化できる可能性がある．

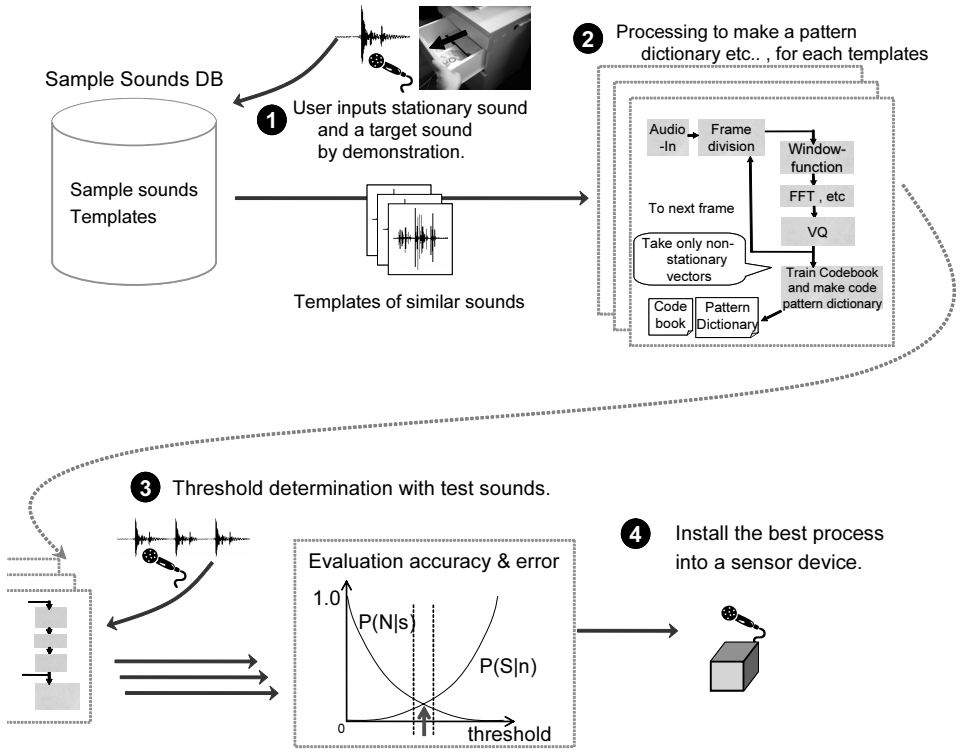

図 4 イベント学習モードにおける処理
Fig. 4 Process in the Event Learning Mode

4.2 サンプル音データベースとパラメータ・テンプレート

3.4 節にて述べたように，未知の生活音・環境音に対する認識処理を生成するためには，それに適した特徴量の組み合わせと，信号処理などに関するパラメータを選択する必要がある．全組み合わせを総当たりする方法も考えられるが，表 3 に示すパラメータの組み合わせだけでも，フレーム長を 128 点，256 点，512 点，1024 点の 4 通りとし，特徴量間の重み (W_f, W_p, W_a) を，それぞれ 0.0 ～ 1.0 の範囲にて 0.1 刻みに変化させていった場合だけでも，5324 通りの試行が必要である．さらに，今後のアルゴリズムの追加などを考慮すると，処理時間の面から現実的ではない．

そこで本手法では，あらかじめ，いくつかの生活音・環境音をサンプル音として収集しておき，それぞれのサンプル音に適したパラメータを保有するサンプル音データベースを構築することにした．サンプル音データベースは，収集したサンプル音とそれぞれに対応する処理をテンプレートとして持つ．テンプレートには，その音を認識するのに適したアルゴリズム，パラメータの組み合わせが付与されている．テンプレートは，あらかじめ時間をかけ，全探索などによって事前に作成しておくため，本システムのユーザが行う必要はない．

Sensor Configurator は Event Learning Mode 時に，このサンプル音データベースの中から，ユーザによって与えられた認識させたい音と類似するサンプル音を自動検索し，いくつかのテンプレートを適したコンフィギュレーションの候補として選出する．それぞれのテンプレートにしたがって学習処理を行い，最終的に最良のテンプレートを提示する．それによって，現実的な試行時間内にてイベント学習を完了できる．

なお，事前のテンプレート作成時には，軽量な処理を目指すために，可能な限り，少ない計算量とメモリ消費量になるようにパラメータを付与することが望ましい．例えば，特徴量ベクトルの次元数を小さくすること，コードブックのサイズを小さくすることが挙げられる．

4.3 Instant Learning

サンプル音データベースを利用する場合の Event Learning Mode の処理手順を図 4 と以下に示す．

(1) サンプル音データベースより，ユーザが認識させたい音に適したテンプレートを探すには，認識処理生成時に選択可能な特徴に応じて探索する必要がある．

今回は，3 節にて述べたように特徴として，主に周波数特性と時間的な振幅変化を用いている．そのため，対象イベント音が入力されたならば，音全体の周波数特性と振幅変化特性について，各サンプル音と認識対象音との類似度を求める．今後，選択可能な特徴量を増やしたならば，このテンプレートの探索時に用いる類似度の種類も増やす必要がある．

類似度 sim の計算を式 6 に示す．式中の s_{sample} はサンプル音，s_{target} は対象イベント音を示す添字である．

$$sim\ (s_{sample}, s_{target}) = sim_f\ (s_{sample}, s_{target}) \times sim_a(s_{sample}, s_{target}) \quad (6)$$

$$sim_f\ (s_{sample}, s_{target}) = \frac{\sum_{n=1}^{N} u(n)w(n)}{\sqrt{\sum_{n=1}^{N} u(n)^2 \sum_{n=1}^{N} w(n)^2}} \quad (7)$$

$$sim_a\ (s_{sample}, s_{target}) = \frac{1}{dtwCost(s_{sample}, s_{target}) + 1} \quad (8)$$

周波数特性に関する類似度 sim_f（式 7 ）は，音の全区間を通じた各周波数帯域におけるパワーの相対的変化量を，N 次元ベクトルにし，2 つの音において類似度を比較した値である．ベクトル間の類似度計算には，Vector Space Model[3] を用いる．式中の $u(n)$ は，サンプル音中の全フレームに対して特徴量ベクトルを算出し，n 番目の周波数特性に関するベクトル要素 $Vec(n)$ の最大値と最小値の差を求めたものである．一方，$w(n)$ は，同様の計算を認識対象のイベント音に対して行ったものである．

続いて，2 つの音の振幅波形の類似度 sim_a の計算式を式 8 に示す．振幅特性に関する類似度は，サンプル音と対象音の間にて振幅波形の類似度を DP マッチングにより求めるものである．式中の $dtwCost$ は DP マッチングの結果，得られる経路コストである．コストが小さいほど，振幅値の波形が類似していることを表す．

各類似度は 0.0 ∼ 1.0 に正規化されているため，最終的な類似度 sim も同様の値の範囲になる．また，値が大きい方が類似することを表す．

(2) 上記 (1) にて求めた類似度 sim リストをソートし，上位のテンプレートをいくつか取得する． 取得数は計算時間を考慮し，4 つ程度が望ましい．

そして，それぞれのテンプレートに基づき，認識対象の音に関するコードブックを LBG アルゴリズム[13] により生成し，続いてそのコードブックを用いて，対象音の符号パターンを生成する．

(3) 各候補テンプレートごとに (2) にて生成されたデータを用い，正解音判定を行うための閾値を決定する．まず，正解データである認識対象の音を，(2) にて作成したコードブック・符号パターンを用いて 3 節にて述べた認識処理を行い，DP マッチングの結果である最小経路コストを算出する．これを数回繰り返し，正解音を入力した際の DP マッチング結果の類似度の分布を得る．続いて，不正解音として，サンプル音データベース中の音を用い，認識対象外の不正解音を入力した際の DP マッチングの出力値の分布を得る．

そして，上記の 2 つの分布より，DP マッチングの最小経路コストに対する閾値を変数とすると，閾値ごとに，以下に示す不正解データ受理率 $P(S|n)$ と正解データ棄却率 $P(N|s)$ が得られる．閾値と，$P(S|n)$ および $P(N|s)$ の関係をグラフにすると，図 4 の (3) のようになる．したがって，$P(S|n) = P(N|s)$ となる点における閾値を最良とする．

$$P(S|n) = \frac{Accepted\ Incorrect\ Sounds}{Total\ of\ Incorrect\ Sounds} \quad (9)$$

$$P(N|s) = \frac{Rejected\ Correct\ Sounds}{Total\ of\ Correct\ Sounds} \quad (10)$$

(4) 最後に，各候補テンプレートにおける (3) までのコンフィギュレーションに基づき，認識処理の認識率，誤認識率を評価する．誤認識率と認識率の評価は (3) にて決定された閾値によって，得られる $P(S|n)$ および $P(N|s)$ を用いる．

そして，もっとも低い誤認識率を持つテンプレートに基づく認識処理を，Sound Sensor Device にインストールする．

以上のようにして本手法では，信号処理プログラミングを用いることなく，柔軟に実世界における音イベントの認識処理の作成支援を実現する．

5. PC におけるプロトタイプ実装

4 節の設計に基づき，Windows PC 上に Instant Learning Sound Sensor のプロトタイプを実装した．

5.1 サンプル音の収集

まず，Sensor Configurator にて用いるサンプル音データベースを構築した．サンプル音としては，現時点では表 3 に示すような 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，サンプリング周波数を 32kHz，サンプリングビットを 16bit にて収集している．収録方法としては，図 1 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，録音した．

そして，各サンプルのイベント音ごとに，適した認識処理，パラメータ設定を人手によって付与し，テンプレートを作成した．この際，計算量とメモリ消費量の軽減のために，少ない特徴量ベクトルの次元数，少ないコードブックの代表ベクトル数，適度な長さのフレーム長 (最小 128 点，最大 512 点) になるように留意した．

5.2 Sensor Configurator

続いて，前節にて述べたサンプル音データベースを用いて実装した Sensor Configurator のスクリーンショットを図 5 に示す．コードブック作成に用いる LBG アルゴリズムの処理には，Speech Signal Processing Toolkit(SPTK)[15] を使用した．

図中の GUI 中央には，定常音 Wave ファイル，対象イベント音 Wave ファイル指定するためのテキストボックスを備える．GUI 下部には，サンプル音に対応するテンプレートを選択可能なリストボックスを備える．通常，このテンプレート選択は，Wave ファイル名が指定された時点で，Sensor Configurator が入力 Wave ファイルと全サンプル音を比較し，自動的に最適なテンプレートが選択されるため，ユーザは操作する必要がない．

使用方法としては，図中のように，(1) ユーザはあらかじめ認識したいイベント音を最大 5 つ，個別に Wave ファイルとして録音を行う．(2) ユーザは定常音とイベント音 Wave ファイルをテキストボックスに指定し，(3) 自動的に選択されたテンプレートを確認後，(4)GUI 上部の Instant Learning ボタンを押すだけである．最後に，出力された最適パラメータ集合であるコンフィグファイル，および，対象イベント音のコードブック・符号パターン辞書，認識プログラム本体を Sound Sensor ソフトウェアを動作させる端末上にコピーして完了する．

5.2.1 Sound Sensor

実際にイベント音を認識処理を行う Sound Sensor を Windows XP 上に C 言語を用いて実装した．本実装は，認識プログラム本体と，その開始・停止を制御する GUI 部分から構成される．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 6 に示す．イベントを検出時には，IP ネットワークを介し，イベント通知の UDP パケットを送信する．

5.3 Sound Recognition Toolkit

Sound Sensor ソフトウェアとユーザが開発するアプリケーションやシステム側にて，イベント通知パケットを受信するための Sound Recognition Toolkit API ライブラリを Windows 用 DLL として実装した．図 7 に本 ToolKit の利用構成例を示す．

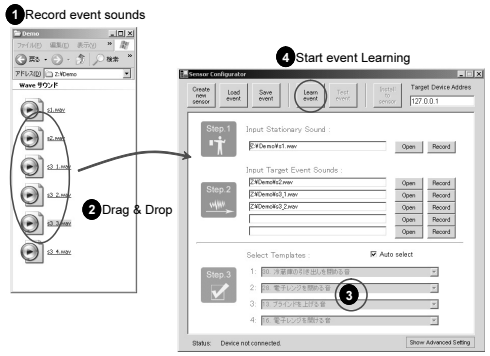

図 5 Sensor Configurator のスクリーンショット
Fig. 5 Screenshot of Sensor Configurator

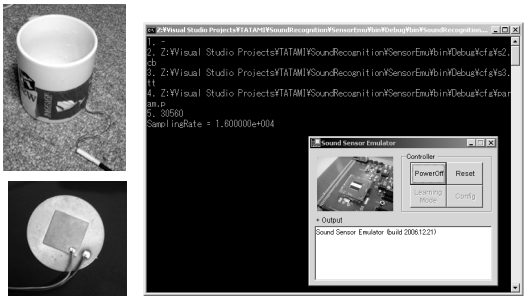

図 6 PC 上の Sound Sensor のスクリーンショット
Fig. 6 Screenshot of Sound Sensor on PC

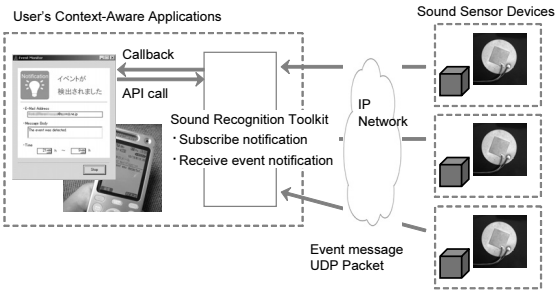

図 7 Sound Recognition Toolkit の利用構成
Fig. 7 Overview of use of Sound Recognition Toolkit

6. 評　　価

前節で実装した PC 上でのプロトタイプを用い，いくつかのイベント音について認識率，誤認識率を評価した．

まず，サンプル音データベースに格納されているサンプル音と，それに対応するテンプレートを用いた場合について評価し，その後，それらテンプレートを用いた Instant Learning によって生成される未知のイ

表 3 サンプル音認識処理の評価結果
Table 3 Result of evaluation with optimized templates

イベント名	認識率 (Recognition rate)	誤認識率 (Error rate)
1. 水道から水が流れる音	100%	0.0%
2. IH クッキングヒータの加熱動作音	100%	0.83%
3. 引き戸を閉める音	100%	1.67%
4. 冷蔵庫の扉を閉める音	100%	2.50%
5. 冷蔵庫の引き出しを開ける音	100%	5.00%
6. 電子レンジ扉を開ける音	100%	13.3%
7. 金属板を叩く音	100%	15.0%
8. ブラインドを上げる 音	100%	15.8%
9. 押しスイッチを押す音	100%	48.3%
10. ホワイトボードに書き込む音	100%	94.2%
11. 金属板を弱く叩く音	98.7%	20.8%
12. 木材机を叩く音	98.1%	31.7%
13. プラスティック板を叩く音	96.3%	17.5%
14. 冷蔵庫の扉を強く閉める音	93.75%	42.5%
15. ドアノブを回し，ドアを開ける音	88.9%	0.0%
16. コーヒーカップにお茶を注ぐ音	86.8 %	5.00%
17. 電話の受話器を下ろす音	86.4%	5.83%
18. クリップボードにペンで書き込む音	85.63%	65.8%
19. ホワイトボード・クリーナーの音	83.4%	79.2%
20. 引き出しを開ける音	83.3%	0.0%
21. AC タップにコンセントを差し込む音	83.3%	0.0%
22. ガラス窓の鍵を開ける音	80.0%	0.0%
23. 冷蔵庫の扉を静かに閉める音	78.6%	35.8%
24. ガラス窓を閉める音	42.2%	0.0%

表 4 テンプレートにより自動生成された未知イベント音認識処理の評価結果
Table 4 Result of evaluation using the instant learning

イベント名	認識率 (Recognition rate)	誤認識率 (Error rate)
1. キーボード・タイピング音	100%	35.0%
2. ブラインドを開ける音	96.9%	0.0%
3. ホワイトボード・クリーナーの音	93.9%	99.2%
4. クリップボードにペンで書き込む音	85.63%	35.8%
5. 引き出しを開ける音	83.3%	1.7%

ベント音に関する認識処理について評価した.

6.1 最適化済みテンプレートの評価

サンプル音データベースに格納されているサンプル音の中から 24 種類の音に対して，次のような評価を行った.

6.1.1 手　　順

(1) 認識率の評価：それぞれのサンプル音のテンプレートに基づいた認識処理において，約 50 回分の正解音を与え，正しく検出された回数を測定し，認識率を算出した.

$$Recognition\ rate = \frac{accepted}{correct} \tag{11}$$

(2) 誤認識率の評価：サンプル音データベース中の 24 種類のサンプル音を，それぞれ 5 回，合計 120 回分の不正解音を与え，誤検出された回数を測定し，誤認識率を算出した.

$$Error\ rate = \frac{accepted}{incorrect} \tag{12}$$

6.1.2 結　　果

結果を表 3 に示す．今回の実装においては，各テンプレートは全探索による最適解ではなく，人手によって付与されている．そのため，必ずしも最適なものとは言えないが，概ね 83% ～ 100% の認識率にて，正解データを認識可能であることが確認できた.

いくつかのテンプレートに関しては，誤認識率が高く，識別が難しい音もあるが，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をあったことに起因する場合が多かった.

また，誤認識率の結果は，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．

6.2 未知の音に対する Instant Learning の評価

続いて，前節のテンプレートを用い，未知のイベント音 5 種類に対して，Sensor Configurator を用い，本研究の提案である Instant Learning により認識処理を生成した．そして，それぞれの音に対して，次のような評価を行った．

6.2.1 手　　順

(1) それぞれの未知のイベント音 5 回分を Sensor Configurator に与え，それぞれの認識プログラムとコンフィグファイルを得た．なお，一部の学習対象音に，サンプル音データベースからの音を利用した．サンプル音に対して Instant Learning を行う際には，交差検定として，そのサンプル音に対応するテンプレートを除去して以後の学習処理を実行した．

(2) 認識率の評価：それぞれの自動選択されたテンプレートに基づいた認識処理において，約 50 回分の正解音を与え，正しく検出された回数を測定し，認識率を算出した．

(3) 誤認識率の評価：サンプル音データベース中の 24 種類のサンプル音を，それぞれ 5 回，合計 120 回分の不正解音を与え，誤検出された回数を測定し，誤認識率を算出した．

6.2.2 結　　果

結果を表 4 に示す．前節の結果と同様に，概ね 80% を超える認識率にて正解データを認識可能であることが確認できる．

今回，Instant Learning によって生成された認識処理は，類似音に適したテンプレートに基づきコンフィギュレーションされている．そのため，必ずしも最適なパラメータ設定とは言いきれないが，認識率 80% ～ 100% と良好な結果が得られた．誤認識率に関しては，ホワイトボード・クリーナーの使用音以外のにおいて，約 35%以下に抑えられている．これより，あらかじめ適したパラメータ設定のされている類似音のテンプレートを用いた自動パラメータ設定の有用性を確認できた．

ホワイトボードクリーナーに関する音の認識が難しかった理由としては，そもそもの音の振幅が非常に小さく，定常時の音と区別が難しかったためと考えられる．このような音への対応としては，今回使用したテンプレートには含まれない別のパラメータ設定を用いること，もしくは，全く別の特徴量抽出アルゴリズムを追加するなどの対策が必要である．

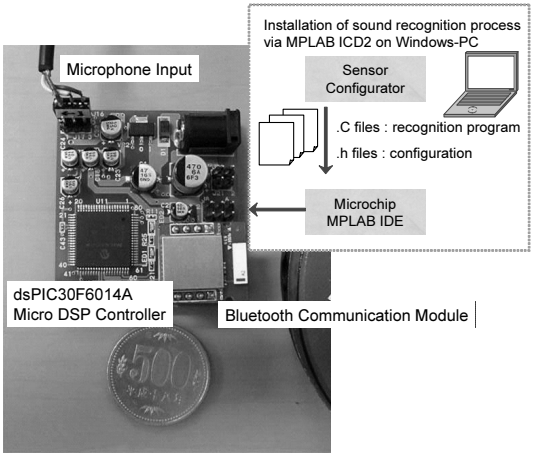

図 8　試作中の dsPIC を用いた Sound Sensor デバイス
Fig. 8　Prototype of Sound Sensor Device with dsPIC

7. 小型・低コストデバイスにおけるプロトタイプ

5 節において，PC 上のプロトタイプについて述べ，6 節にて評価を行い，Instant Learning の有効性について確認した．次の段階として，我々は小型・低コストな Sound Sensor Device のプロトタイプを実装し，小型・低コストデバイス上での実現可能性について検討を行った．

本プロトタイプで実装した小型・低コストなデバイスによる Sound Sensor Device の概要は，次の通りである．低コストな DSP マイクロコントローラとして，Microchip 社の dsPIC30F6014A（8KB Data RAM）を用いた．また，音信号は Silicon Laboratories 社の Si3000 Codec チップを用い，サンプリング周波数 12KHz，サンプリングビット 16 ビットにて取り込みを行う．イベント通知のための通信は，Parani 社の ESD200 を用い，Bluetooth 経由にて行う．イベント音認識プログラムをマイクロコントローラへ書き込み際は，Sensor Configurator より出力されたコードブック，符号パターン，パラメータ設定などを dsPIC 向け C コンパイラのヘッダファイルとして変換し，あらかじめ作成済みの認識処理本体のソースファイルと組み合わせ，Microchip 社の MPLAB IDE を経由してコンパイル，書き込みを行う．

続いて，実際にサンプルとして収録したいくつかの音を用いて，計算量とメモリ消費量を評価した．イベント音認識をリアルタイムにて処理するためには，フレーム毎の処理時間を $FrameShiftLength/SamplingFrequency$ 以下に収める必要がある．例えば，図 1 および表

２中の水道が水が流れる音 (a) では，5msec(= 60/12) である．この例のパラメータ設定では，シミュレーションを用いフレーム毎の処理時間を計測したところ，システムクロックを 58.98MHz（14.74MIPS）に，DP マッチング時の比較符号パターンの長さを 60 とした時に平均 3.49$msec$ であり，条件を満たす．

メモリ消費量に関しても，処理中に使用される RAM サイズが 4KB 未満であることを確認できた．イベント音認識処理プログラム本体のコードサイズは，コードブックなどの定数データを含め，約 10KB であった．

他にも，引き出しを開け閉めする音についても，実際に本センサボードを用いてリアルタイム処理可能なことを確認した．

本プロトタイプの実装により，今回設計した音認識処理を用いた小型・低コストな Sound Sensor Device の実現可能性について確認できた．

8. ま と め

本稿では，実世界における生活音や環境音を，小型・低コストなセンサデバイスより，コンテクスト情報として手軽に利用可能にすることを目的とし，認識対象音を即興的に学習可能な Instant Learning Sound Sensor を提案した．そして，その設計と，PC および低コスト DSP マイクロコントローラ上におけるプロトタイプの実装について述べた．

提案システムでは，イベント音の認識処理を生成する際，対象音ごとに適した特徴量抽出処理や認識アルゴリズムを自動的に選択し，ユーザに提示する．このように，対象イベントに認識処理を特化することにより，小型・低コストデバイス上での動作を実現した．プロトタイプ実装を通じて，実際にマイクロコントローラ上においても，動作可能なことを確認した．

提案システム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用いることなく，実世界の生活音や環境音認識をイベントとして取得可能なアプリケーションやコンテクストアウェア・システムを手軽に構築することができる．

今後の課題は次の通りである．現在の Sensor Configurator では，サンプル音データベースと各サンプル音に適したパラメータをテンプレートとして用い，未知のイベント音に対するパラメータ決定を行っている．これは，学習時にかかる処理時間を軽減したいという目的より検討した手法であるが，焼き鈍し法や遺伝的アルゴリズムといった他の最適解探索手法の適用についても検討していきたい．加えて，さらに多くの生活音や環境音の収集，選択可能なフィルタや認識処理の追加を行う予定である．また，学習操作にかかるユーザの負担軽減など，ユーザビリティに関する評価も検討していきたい．

参 考 文 献

1) Crossbow Technology, MOTE, http://www.xbow.com/Products/Wireless_Sensor_Networks.htm
2) C. S. Myers and L. R. Rabiner.: A comparative study of several dynamic time-warping algorithms for connected word recognition, IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. 29, No. 3, pp.351–363 (1981)
3) G. Salton and A. Wong and C. S. Yang.: A vector space model for automatic indexing, Communications of the ACM, Vol.18, No.11, pp.613–620 (1975)
4) H. Sakoe, S. Chiba.: A Dynamic Programming Algorithm Optimization for Spoken Word Recognition, IEEE Trans. Acoustics, Speech, and Signal Processing, Vol. 26, No. 1, pp.43–49 (1978)
5) Jianfeng Chen, Alvin Harvey Kam, Jianmin Zhang, Ning Liu, Louis Shue : Bathroom Activity Monitoring Based on Sound, Pervasive 2005, pp.47–61 (2005)
6) Ling Bao, Stephen S. Intille.: Activity Recognition from User-Annotated Acceleration Data, Pervasive 2004, pp.1–17 (2004)
7) Ling Ma, Dan Smith, Ben Milner.: Environmental Noise Classification for Context-Aware Applications, Database and Expert Systems Applications, pp.360–370 (2003)
8) Microchip Technology Inc, dsPIC Digital Signal Controllers, http://www.microchip.com
9) Paul Lukowicz, Jamie A. Ward, Holger Junker, Mathias Stager, Gerhard Troster, Amin Atrash, Thad Starner.: Recognizing Workshop Activity Using Body Worn Microphones and Accelerometers, Pervasive 2004, pp.18–32 (2004)
10) Robert M.Gray.: Vector Quantization, IEEE ASSP Mag., 1, 2, pp.4–28 (1984)
11) Smart-Its Project, Emmanuel Munguia Tapia, Stephen S. Intille, Louis Lopez, and Kent Larson.: The design of a portable kit of wireless sensors for naturalistic data collection, Pervasive 2006, pp. 117–134 (2006), (http://www.smart-its.org/)
12) Susan Cotterell, Ryan Mannion, Frank Vahid, Harry Hsieh.: eBlocks - An Enabling Technology for Basic Sensor Based Systems, IPSN

Track on Sensor Platform, Tools and Desing Methods for Networked Embedded Systems (SPOTS), (2005)
13) Yoseph Linde, Andres Buzo, Robert M.Gray.: An Algorithm for Vector Quantizer Design, IEEE Trans.Commun.,COM － 28, 1, pp.84–95 (1980)
14) 井上靖浩, 若原淳, 柳生辰夫, 高浜盛雄：HMM による生活音分析の考察, 計測自動制御学会システム, (2004)
15) 音声信号処理ツールキット (SPTK), http://kt-lab.ics.nitech.ac.jp/tokuda/SPTK/index-j.html
16) 佐藤誠, 森田千絵, 土井美和子：生体データと加速度データを用いた行動認識, 情報処理学会第６５回全国大会, pp.5-239–5-242 (2003)
17) 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 VolJ82-D-II No.4, pp.721–731 (1999)
18) 古井貞熙：音声情報処理, 森北出版, (1998)