平成27年度

筑波大学情報学群情報科学類

卒業研究論文

題目
スマートホームユーザ向けの
イベント学習・認識モジュールの開発

主専攻　　情報システム主専攻

著者　　石田 隼己

指導教員　高橋伸　志築文太郎　田中二郎

要　旨

スマートホーム向けのアプリケーションを自分の手で作成できる製品が次々と登場している．現在こういった製品でスマートホームアプリケーションを手軽に作成するためには，センサ値などを意識することなく,「何が起こったか」というイベントを入力として利用できることが望まれる．そのためイベントの認識が必要だが，入力として利用したいイベントは人によって異なる．そこで，本システムではイベントの認識だけでなくユーザ自身でイベントを学習させることができるようにした．これにより，ユーザは自身で学習させたイベントの発生をトリガとしたスマートホームアプリケーションの作成ができる．学習させる操作やアプリケーションの作成にはスマートフォンを利用する．本システムは音の発生を伴うイベントを対象とし，センサとしてマイクを利用している．

試用評価の結果，ユーザが学習させたイベントを認識し，それをトリガとしてスマートホームアプリケーションの作成ができることを確認した．

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景

さまざまなものがインターネットに繋がる，Internet of Things(IoT) という概念が広まっており，盛んに研究が行われたり様々な製品が登場したりしている [1]．産業分野では，小型のセンサと無線機を産業機械に搭載して動きをモニタリングしたり，畑に設置して温度などの情報を取得したりといった形で活用されている．また，これらの技術が徐々にスマートホーム製品などとして徐々に家庭向けに広がり始めている [2]．特に，最近は気軽に IoT デバイスを制御し，スマートホームアプリケーションを自分の手で作成 (DIY:Do It Yourself) できる製品が次々と登場している．例えば，SONY の MESH[3] と呼ばれる製品がある．これは，モジュール化されたセンサやアクチュエータをユーザが好きな場所に設置し，スマートフォンアプリから GUI を通じて動作設定をすると，センサの反応に応じてアクチュエータを動作させることができる製品である．作例としては，振動を検知するモジュールを壁に設置し，壁をノックすると離れた場所に設置した LED モジュールの LED が点灯して人を呼び出せるといったものがある．

しかし，現在多くの製品でユーザがアプリケーションを作成する際に，センサ値を直接利用したり，センサが反応するしきい値を調整したりする必要がある．スマートホームアプリケーションをより手軽に作成するためには，センサ値などを意識することなく,「何が起こったか」というイベントを入力として利用できることが望まれる．そのためには，イベントの認識が必要だが，入力として利用したいイベントは人によって異なる．どんなイベントを認識させるかユーザが自由に設定できる必要があるが，全てのユーザの欲求を満たすイベントを予め認識できるようにすることは不可能である．

1.2　目的

本研究の目的は，ユーザ自身の手によってイベントを学習させ，認識できるようにすることである．認識させたいイベントをユーザ自身の手によって学習させることで，どんなイベントを認識させるかユーザが自由に設定できるようになる．イベントの認識ができるようになることで，センサ値などを意識することなく，イベントを入力として利用できるようになる．さらに，イベントの学習・認識を可能にするだけでなく，認識したイベントの発生をトリガとしてスマートホームアプリケーションの作成に利用できるようにする．また，ユーザが DIY すること考慮し，導入するところから実際に利用するところまで，専門家のアドバイ

ス無く利用できるようにする．

1.3 アプローチ

前節の目的を達成するために，イベント学習・認識モジュール「Event Learning Sensor」を開発する．Event Learning Sensor でのイベント認識には音を利用する．音を利用することで，ドアを開けるといったユーザの行動や，洗濯機のブザーのような家電の動作をイベントとして認識できる．そして，認識したイベントの発生を入力として，スマートホームアプリケーションの作成に利用できるようにするために，既存のスマートホーム製品との連携を行う．Event Learning Sensor の操作やスマートホームアプリケーションの作成は全てスマートフォンから行えるようにする．これにより，スマートホームアプリケーションの作成がユーザ自身の手でできるようになる．

1.4 構成

本論文の構成について述べる．本章では，本研究の背景について述べ，それを踏まえて本研究の目的とアプローチについて述べた．続いて第 2 章では，本研究に関連する研究について述べる．第 3 章では，本研究にて開発を行う Event Learning Sensor の概要や利用シーンについて解説を行う．第 4 章では，Event Learning Sensor の実装について詳細に述べる．第 5 章では，ユーザによる試用評価によって Event Learning Sensor を評価し，考察する．第 6 章では，本論文の結論と今後の課題を述べる．

第2章　関連研究

2.1　スマートホームに関する研究

スマートホームに関する研究が長く行われているのにもかかわらず，ホームオートメーション製品がなかなか一般に普及していないということについて調査した研究がある [4]．この研究によって，ホームオートメーション製品の普及への障害が 4 つあることが明らかになった．所有するのに高いコストがかかること，柔軟性が欠如していること，管理性が乏しいこと，セキュリティの担保が難しいことである．本研究では，一般に市販されている部品を用いて，なるべく低コストに抑えるような実装を行った．また，ユーザ自身でイベントを学習させることができるため，ユーザの幅広い欲求に応えることが可能である．

既存の DIY スマートホーム製品について，ユーザ調査を行った研究がある [5]．DIY スマートホーム製品に興味を持っている被験者を集め，利用状況を観察した．実験の結果，評価を行ったすべての家庭で，自作したアプリケーションが活用されていることが確認できた．このことから，DIY スマートホーム製品は有用であり，今後普及していくと考えることができる．本研究では，こうした DIY スマートホーム製品をより使いやすいものにすることを目的としている．

DIY スマートホーム製品のアプリケーション作成の UI として，トリガアクションプログラミングという手法がある [6]．トリガアクションプログラミングは,「何がどうなったら，何をどうする」というような形で，トリガとアクションを指定するだけでプログラミングができるようになっている．そのため，プログラムに関する知識が一切必要なく，プログラミングを習っていないユーザでもすぐにアプリケーション作成ができるという点が特徴である．本研究でのスマートホームアプリケーションの作成にも，トリガアクションプログラミングを利用する．

2.2　行動認識に関する研究

大内らによる携帯電話の加速度センサと音センサを利用して生活行動認識を行うという研究がある [7]．この研究では，計算コストのかかる音認識処理を常時行うことは避け，携帯電話に内蔵されている加速度センサでユーザの大まかな行動認識を行った後，その結果に応じてマイクを起動し，環境音の分析によって詳細な行動認識を行っている．本研究では，Event Learning Sensor は動かないため加速度センサを用いることは出来ないが，しきい値を超える比較的大きな音の発生があった場合にのみ，認識処理を行うようにしている．

室内に設置し，生活行動の認識を行う研究としては，熱センサアレイを利用したものがある [8]．この研究では，天井に 8 × 8 の熱センサアレイを設置し，温度変化の勾配や最高温度などから，屋内の様々な機器の動作状況や，人の活動を認識している．本研究ではセンサにマイクを利用するが，音だけでは認識できないイベントの認識や，複数のセンサを併用することによって認識率を向上させるといったことも想定される．

根岸らによる音のイベント認識を手軽に利用できるようにするセンサモジュールの研究がある [9]．この研究では，信号解析による音イベントの認識を手軽かつ低コストに利用できるスマートセンサと，自動的に最適な認識処理を行う手法を提案している．この研究では，認識精度などについての検証を行っていた．本研究では，イベント認識を手軽に利用できるモジュールを開発するだけでなく，ユーザが実際にアプリケーションを作成し，利用できるようにすることを目標とする．

第3章　Event Learning Sensor

3.1　概要

Event Learning Sensor は，ユーザ自身の手でイベントを学習させ，その学習させたイベントを認識することができるモジュールである．そして，認識したイベントをトリガとして，アプリケーションを作成することができる．作成することのできるアプリケーションは，イベントを認識してスマートフォンに通知を送るといったものから，既存のスマートホーム製品を制御するといったものまでと多岐にわたる．また，ユーザはスマートフォンから Event Learning Sensor の操作や設定，アプリケーションの作成ができる．

3.2　利用イメージ

ドアを開けたら部屋の照明が点灯するというアプリケーションの作成を例にとって，従来のスマートホーム製品の利用イメージと，Event Learning Sensor の利用イメージを示し，比較を行う．

3.2.1　従来の DIY スマートホーム製品の利用イメージ

序論で述べたような従来の DIY スマートホーム製品の利用イメージを示す.

1. ドアにセンサモジュール（例えば振動センサモジュール）を，部屋にスマート電球を設置する．
2. スマートフォンとセンサモジュールを接続する．
3. スマートフォンアプリから，センサモジュールが反応するしきい値を設定する．
4. 実際にドアを開けてみてしきい値が適切かどうか調べる．しきい値が低すぎて反応しなかったり，逆にしきい値が高すぎて誤反応してしまう場合は，3 に戻る．
5. センサが反応したことをトリガとして， スマート電球が点灯するようにスマートフォンから設定する．
6. ドアを開けるとセンサが反応し， スマート電球が点灯するようになる．

3.2.2 Event Learning Sensor の利用イメージ

Event Learning Sensor を利用した場合の利用イメージを示す.

1. ドア付近に Event Learning Sensor を，部屋にスマート電球を設置する.
2. スマートフォンと Event Learning Sensor を接続する.
3. スマートフォンから操作して，Event Learning Sensor を学習モードにする.
4. 実際にドアを開けるというイベントを発生させ，Event Learning Sensor に学習させる.
5. ドアを開けたことをトリガとして， スマート電球が点灯するようにスマートフォンから設定する.
6. ドアを開けるとそのイベントを認識し， スマート電球が点灯するようになる.

このように，ユーザは Event Learning Sensor を使うことで，しきい値の調整といったような面倒な作業を行うこと無くスマートホームアプリケーションを作成することができるようになる．また，従来の DIY スマートホーム製品は 1 つしきい値を設定してしまったら，1 つのセンサモジュールで 1 つのイベントしか認識することが出来ない．しかし，Event Learning Sensor は複数のイベントを学習・認識できるため，1 つのモジュールで複数のイベントを認識することができるようになる.

第4章　実装

4.1　設計

Event Learning Sensor を利用して既存のスマートホーム製品を制御するためのシステム全体構成を図 4.1 に示す.

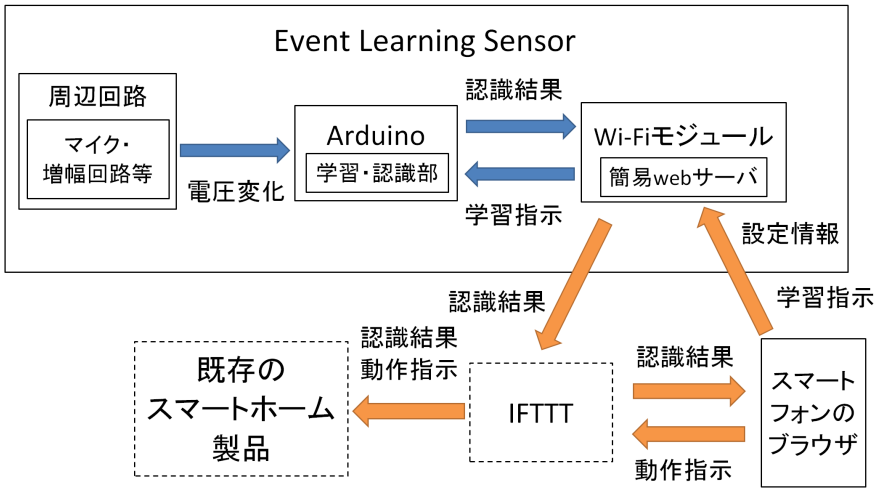

図 4.1: システムの全体構成

利用イメージを実現するための Event Learning Sensor の設計を行う．Event Learning Sensor を構成する要素として，イベントの学習・認識を行う「学習・認識部」，どんなイベントが発生しているのかをセンシングする「センサ部」，スマートフォンに通知を送ったり，既存のスマートホーム製品を制御するための「通信部」が必要であると考えられる．また，Event Learning Sensor から様々な既存のスマートホーム製品を制御したいが，多くの製品が存在する．そのため，いくつもの製品との連携を実現するのは非常に困難である．そこで，さまざまなサービス同士を連携させることのできるクラウドサービスを利用する．

イベント認識に音を利用するため，センサ部はマイクと信号を増幅するための増幅回路から構成される．それに加えて，Event Learning Sensor の周辺回路としてユーザに動作状況を知らせるための LED 等も必要となる．

学習・認識部には，センサ等が扱いやすく，プロトタイプ実装を行うのに適している Arduino を利用する．学習・認識には，マイクから取得した信号に対して FFT を行い，その結果得ら

れた振幅スペクトルを利用する．

通信部には Wi-Fi モジュールを利用する．Bluetooth も検討したが，Bluetooth ではクラウドサービスと通信する場合に，通信を中継するスマートフォン等が常に必要となってしまう．しかし，Wi-Fi ならば学習の指示を出すときなど，必要なときにスマートフォン等があればよく，クラウドサービスとの通信は単独で行える．

ただし，Wi-Fi は利用前にアクセスポイントの設定が必要である．UI の限られた Wi-Fi デバイスの接続設定の手法を検討した Michae らによる研究がある [10]．この論文では，Wi-Fi の接続に必要なアクセスポイントの情報をスマートフォンから送信するための手法をいくつか実装し，ユーザ評価を行っていた．その結果，Web ブラウザを用いて Wi-Fi デバイスにアクセスしてアクセスポイントの情報を送信する手法と，オーディオケーブルを用いて Wi-Fi デバイスと接続して情報を送信する手法の成功率が高いということがわかった．どちらの手法も，スマートフォンの OS を問わずに利用することができるという特徴がある．しかし，Web ブラウザを用いる手法は追加のハードウェアが不要だが，オーディオケーブルを用いる手法はケーブルや復調器が必要となる．そのため，本研究では Wi-Fi モジュールのアクセスポイント設定と，Event Learning Sensor をスマートフォンから操作する手法として，web ブラウザを利用する．

クラウドサービスについても検討する．さまざまなサービス同士を連携させるサービスで，代表的なものに IFTTT[11] というサービスがある．今回はこの IFTTT を利用する．IFTTT を利用して連携させることのできるサービスは，Twitter などに代表される SNS から，メールサービス，スマートフォンの状況，そしてスマートホーム製品まで多岐にわたる．IFTTT ではこれらのさまざまなサービスを，関連研究で述べたトリガアクションプログラミング方式で設定することにより，連携させることができる．例えば，Google カレンダーと Twitter を連携させて予定があったら自分自身にツイートをして知らせるといったような，他のサービスとまたがったアプリケーションが簡単に作成できる．

4.2 プロトタイプ実装 (1)

設計を元に，Event Learning Sensor の実装を検討するため，ブレッドボード上で Event Learning Sensor のプロトタイプを作成した．このブレッドボード上での実装を元に試用評価等を行い，そこで得られたフィードバックをプロトタイプ実装 (2) に反映させる．

4.2.1 開発環境

本研究では，学習・認識部と Wi-Fi モジュールのプログラムを作成する．学習・認識部は Arduino で作成するため，ArduinoIDE1.6.6 を利用した．また，Wi-Fi モジュールも ArduinoIDE の追加ボードとして利用できるパッケージが存在するため，ArduinoIDE1.6.6 を利用して作成した．

4.2.2　ハードウェア実装

Event Learning Sensor のハードウェア実装を示す．各部分の実装に利用した主な部品は表 4.1 の通りである．

表 4.1: プロトタイプ実装 (1):使用部品表

使用箇所	名称	メーカー	型番
センサ部	コンデンサマイク	SPL	XCM-6035P
センサ部	アンプ	新日本無線	NJM386BD
学習・認識部	Arduino	Arduino SRL	Arduino UNO R3
通信部	Wi-Fi モジュール	Espressif Systems	ESP-WROOM-02

マイクからの信号はアンプで 200 倍に増幅され，Arduino のアナログ入力にて取得される．なお，電源は PC との USB 接続で Arduino を駆動させ，周辺回路は Arduino の 5V ピンからの電源で駆動させた．Wi-Fi モジュールは動作電源範囲が 3.0-3.6V のため，3.3V の AC アダプタを用いて駆動させた．また，Wi-Fi モジュールにはリセットボタンを取り付けた．プロトタイプ実装 (1) の回路図を 4.2 を図 4.3 に示す．

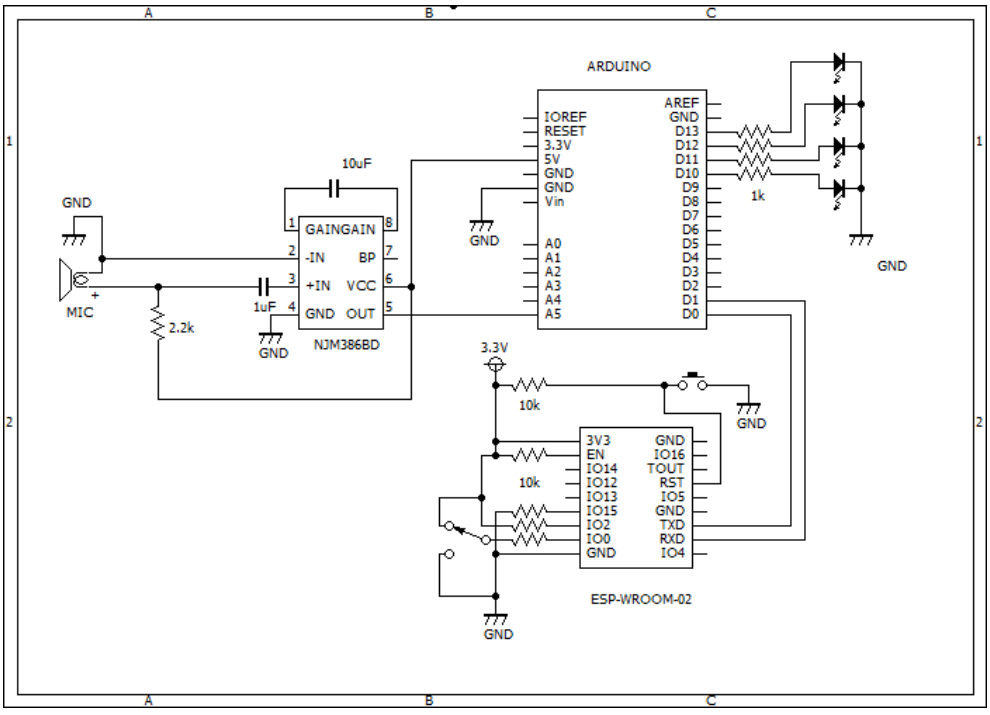

図 4.2: プロトタイプ実装 (1) の回路図

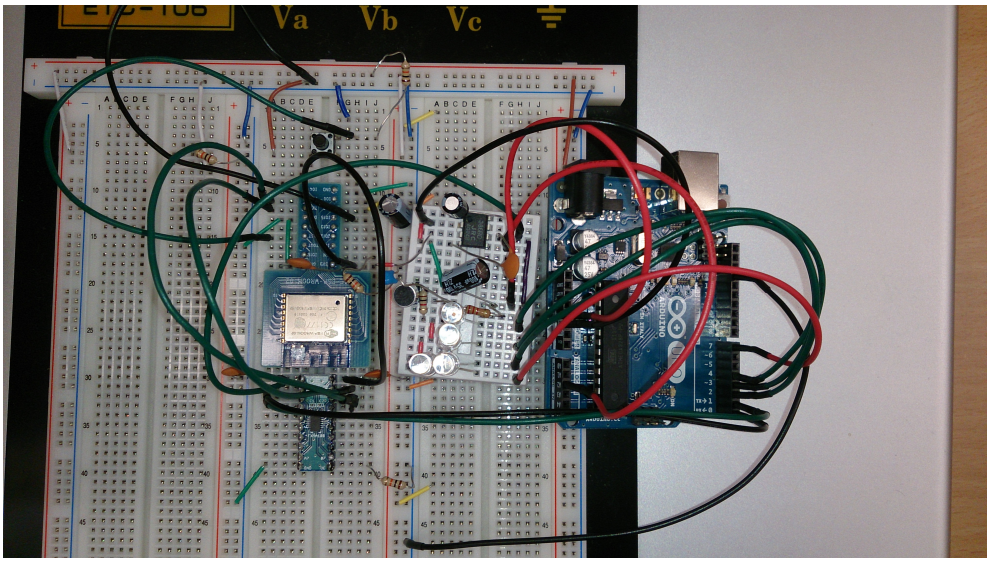

図 4.3: Event Learning Sensor のプロトタイプ実装 (1)

4.2.3　ソフトウェア実装

学習・認識部

学習・認識部のプログラムについて解説する．学習・認識部は，増幅回路によって増幅されたコンデンサマイクからの信号を取得し続け，しきい値を超える音の発生がないか監視する．しきい値を超える大きな音が発生したら，学習の指示が届いているかを確認する．学習の指示は，Wi-Fi モジュールからシリアル通信で送られる．学習の指示が来ていれば学習を行い，来ていなければ認識を行う．

学習は，まずコンデンサマイクからの信号を 10kHz で 256 個サンプリングする．そして，サンプリングしたデータに対して FFT を行う．その結果得られた振幅スペクトルを 0-100 の整数に正規化[1]して Arduino の不揮発性メモリ (EEPROM) に格納する．整数に正規化するのは，EEPROM の容量が 1 キロバイトと限られており，浮動小数点型のデータを格納することができないためである．

認識は，学習と同様にして得た正規化された振幅スペクトルを， 保存されている各イベントの正規化された振幅スペクトルと比較し，各周波数の 2 乗誤差の和を計算し， 最も小さいものを認識結果とする．ただし， 2 乗誤差の和がしきい値より大きい場合は学習したイベントではないものとする．認識の例を図 4.4 に示す．図の波形は正規化された振幅スペクトルを示している．学習済みのイベントが 2 つあるとして，イベント A の学習データを青，B を赤で示す．そして，認識対象として入力されたデータを緑で示している．まずは，入力と学習データ A を各周波数で比較し，差を 2 乗して和を計算する．次に，入力と学習データ B を同

[1]100 という数値に大きな意味はなく，もっと大きな数を使えばそれだけ精度が向上するはずである．

様に比較して和を計算する．図 4.4 の場合は，学習データ B との 2 乗誤差の和の方が小さいため，入力されたデータはイベント B として識別される．今回の実装では，学習・認識できるイベント数を最大 4 つとしてある．認識結果は，シリアル通信によって Wi-Fi モジュールに送信される．

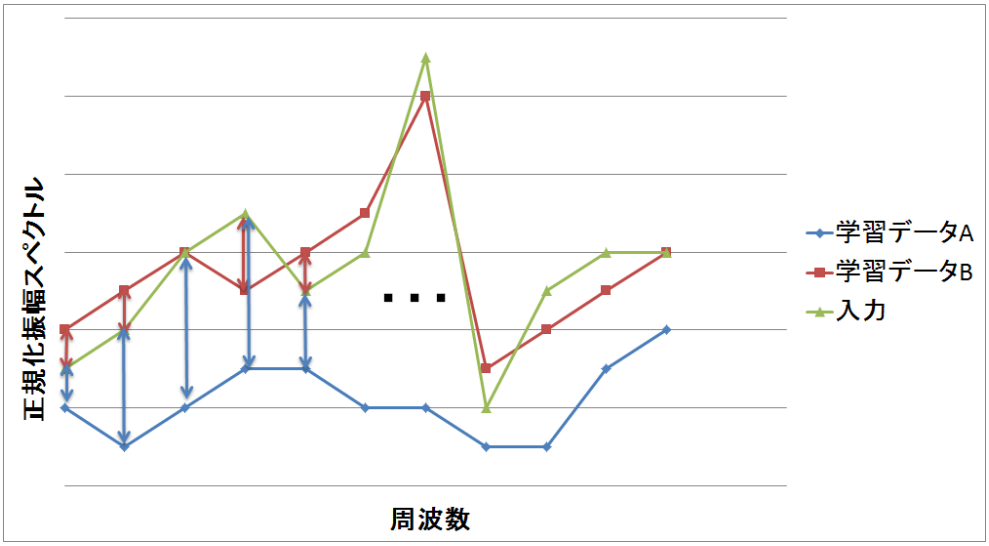

図 4.4: 認識の例

Wi-Fi モジュール

Wi-Fi モジュールのプログラムについて解説する．Wi-Fi モジュールは，起動時に不揮発性メモリ (EEPROM) から Wi-Fi の設定情報を復元し，接続を試みる．初回起動時は，設定情報が存在しないため，復元に失敗する．復元または接続に失敗した場合は，初期設定を行う．初期設定の場合には，Wi-Fi モジュールを子機モードにしてネットワークをスキャンし，接続可能なアクセスポイントの SSID を取得し，保存する．その後，Wi-Fi モジュールを親機モードに変更して web サーバを起動し，スマートフォンからの接続を待つ．スマートフォンを操作して Wi-Fi モジュールのアクセスポイントに接続し，ブラウザから Wi-Fi モジュールの web サーバにアクセスする．すると，スマートフォンに図 4.5(左) のような初期設定の画面が表示される．この時のネットワーク構成を図 4.6 に示す．スマートフォンに表示された初期設定画面で Event Learning Sensor を接続するアクセスポイントを選択してパスワードと IFTTT との連携に必要なキーを入力して送信する．すると，Wi-Fi モジュールには HTTP の GET リクエストとして，スマートフォンで入力したデータが送信されてくる．Wi-Fi モジュールは，このデータを EEPROM に保存する．その後，ユーザはリセットボタンを押し Wi-Fi モジュールの再起動を行う．

Wi-Fi モジュールが無事設定を復元して接続に成功し，初期設定と同様にスマートフォンの

図 4.5: 画面表示の例　(左) 初期設定画面　(右) 設定画面

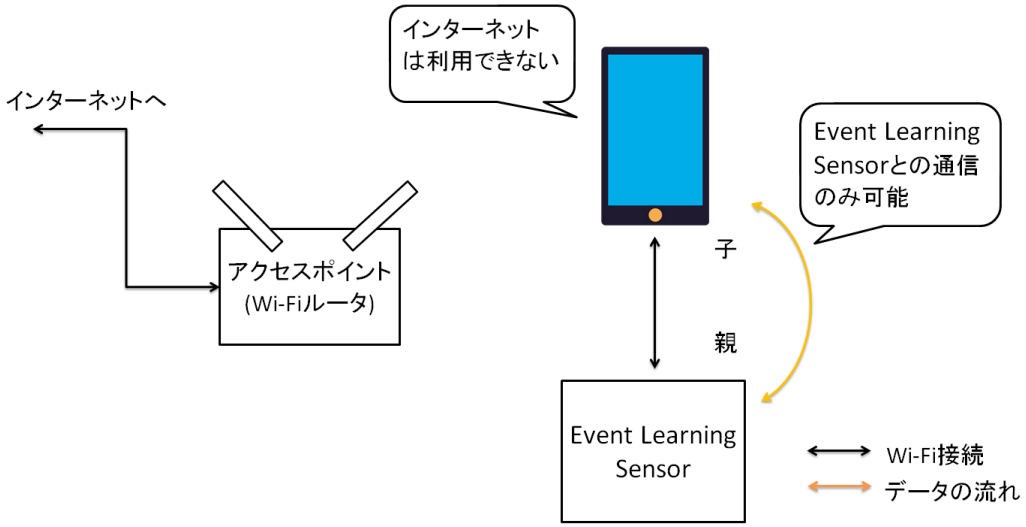

図 4.6: 初期設定時のネットワーク構成図

ブラウザからアクセスすると，今回は Event Learning Sensor の設定画面が表示される．設定画面の例を図 4.5(右) に示す．この設定画面から，学習・認識部に学習の指示を出すことができる．ラジオボタンで 4 つの中からイベント ID を選択し，イベント名を入力して実行を押す．すると，Wi-Fi モジュールにイベント名が記録され，学習・認識部にシリアル通信で学習の指示が送られる．また，この設定画面から Event Learning Sensor の初期化も実施することができる．初期化を押すと，Wi-Fi モジュールの EEPROM に記憶されている全てのデータを消去し，学習・認識部にも学習データをすべて消すようにシリアル通信で命令を送る．なお，初期化を行った場合も手動でリセットボタンを押してリセットを行う必要がある．

Wi-Fi モジュールが無事設定を復元して接続に成功した場合には，Wi-Fi モジュールが親機・子機両方の振る舞いをするようにした．つまり，初期設定で設定したアクセスポイントの子機になりながら，親機としてスマートフォンからの接続も受け付けることができるようになっている．この時のネットワーク構成を図 4.7 に示す．

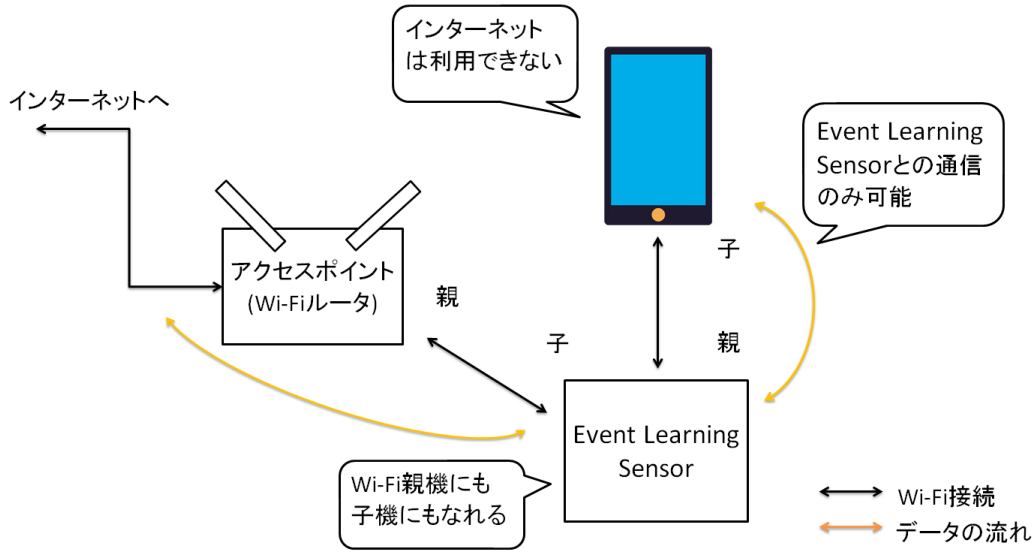

図 4.7: Event Learning Sensor に直接接続したときのネットワーク構成図

設定画面は，Event Learning Sensor に直接接続した場合でも，アクセスポイントを経由して接続した場合でも同じものを表示する．直接接続した場合，スマートフォンからインターネットにアクセスすることが出来ないが，アクセスポイントを経由して接続する場合は，インターネットへのアクセスが可能となる．ただ，アクセスポイントを経由して接続するためには，接続しているアクセスポイントでの IP アドレスが必要となる．そのため，初期設定直後は図 4.7 のようにスマートフォンと Event Learning Sensor を直接接続して Wi-Fi ルータ下での IP アドレスを確認する．それ以降は，アクセスポイントを経由して接続することで，スマートフォンで Event Learning Sensor の設定画面の表示とインターネット接続を両立することができるようになった．スマートフォンと Event Learning Sensor をルータを介して接続し

たときのネットワーク構成を図 4.8 に示す.

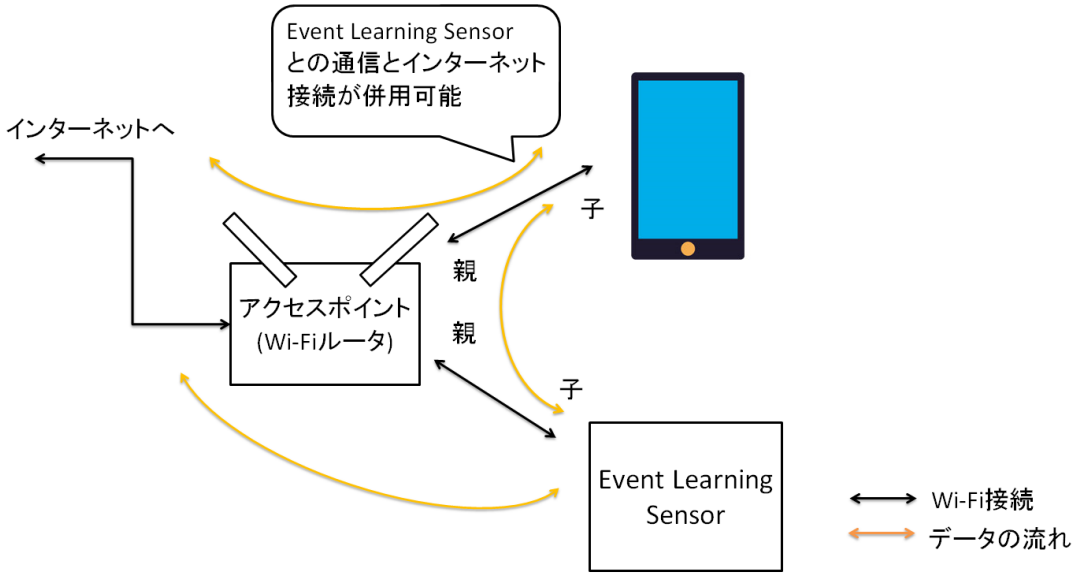

図 4.8: ルーター経由で Event Learning Sensor に接続するときのネットワーク構成図

IFTTT との連携

学習・認識モジュールから認識結果が送られてきた場合には，Wi-Fi モジュールは IFTTT に通知を行う．IFTTT では，作成したアプリケーション（動作設定）を「レシピ」，連携させるサービスを「チャネル」と呼ぶ．IFTTT のチャネルの一つに自作ソフト等とのやり取りに利用できる Maker チャネルがある．Maker チャネルを利用した IFTTT への通知は，HTTP の GET リクエストで実施する．
”`https://maker.ifttt.com/trigger/[`イベント名`]/with/key/[`連携用のキー`]`” へ GET リクエストを送ることで，IFTTT にイベントの発生を知らせる．

4.2.4 試用評価 (1)

プロトタイプ実装 (1) の試用評価を行った．初期設定からアプリケーションの動作を確認するところまでの一連の操作を被験者に行ってもらい，その時の様子を観察し，インタビューを行った．この試用評価でユーザから得られたフィードバックをプロトタイプ実装 (2) に反映する．

被験者

22 歳から 24 歳までの情報系学生 2 名 (ユーザ A，B とする)

タスク

研究室で，プロトタイプ実装 (1) の Event Learning Sensor と被験者自身のスマートフォンを利用して試用評価を行った．また，連携するスマートホーム製品として，スマート電球 Philips Hue[12] を用意した．被験者は，Event Learning Sensor の取扱説明書を見ながら，事前準備から作成したアプリケーションの動作を確認するところまでを実施する．動作の確認が終了したらインタビューを行う．実際に利用することを想定し，被験者にはなるべく説明書のみを参考にして作業するように指示をした．ただし，被験者が途中で困ってしまった場合には口頭で説明を行った．被験者の実験の流れは以下の通りである．

1. 実験内容の説明を受け，取扱説明書を受け取る．
2. 事前準備として IFTTT のアカウントを取得する．
3. Event Learning Sensor の初期設定を行う．
4. Event Learning Sensor の設定画面を表示させる．
5. トリガとしたいイベントを決め，学習させる．
6. 認識したイベントをトリガとしてどんな動作をするか，IFTTT のアプリから設定する．
7. 設定したとおりに動くことを確認する．
8. 感想や改善してほしい点があれば伝える．

結果

試用してもらった結果，被験者が実際に作成したアプリケーションを紹介する．ユーザ A は,「掃除機を使ったらメールで通知を送る」というアプリケーションを作成した．ユーザ B は「手を叩いたらスマート電球の on/off が切り替わる」というアプリケーションを作成した．ユーザ B の試用評価の様子を 4.9 に示す．アプリケーションを利用するまでにかかった時間は，IFTTT のアカウント取得から，アプリケーションの動作確認ができるまでで，ユーザ A が約 40 分，ユーザ B が約 35 分だった．

被験者の様子を観察した結果， 研究概要で示した利用イメージを一通りこなせることが確認できた．ただ，試用評価中に動作が不安定でうまくページが表示されないなどのトラブルが発生し，どちらの被験者にも口頭での指示を行ったり，一旦リセットボタンを押して様子をみたりした．

インタビューの結果，どちらの被験者からもシステムの動作状況， 特に学習しているタイミングがわからないという意見が得られた．さらに，音を拾える範囲がわからないという意見や， 認識してから設定した動作が行われるのに時間がかかるという意見も得た．また，説明書の不備や改善点も指摘された．

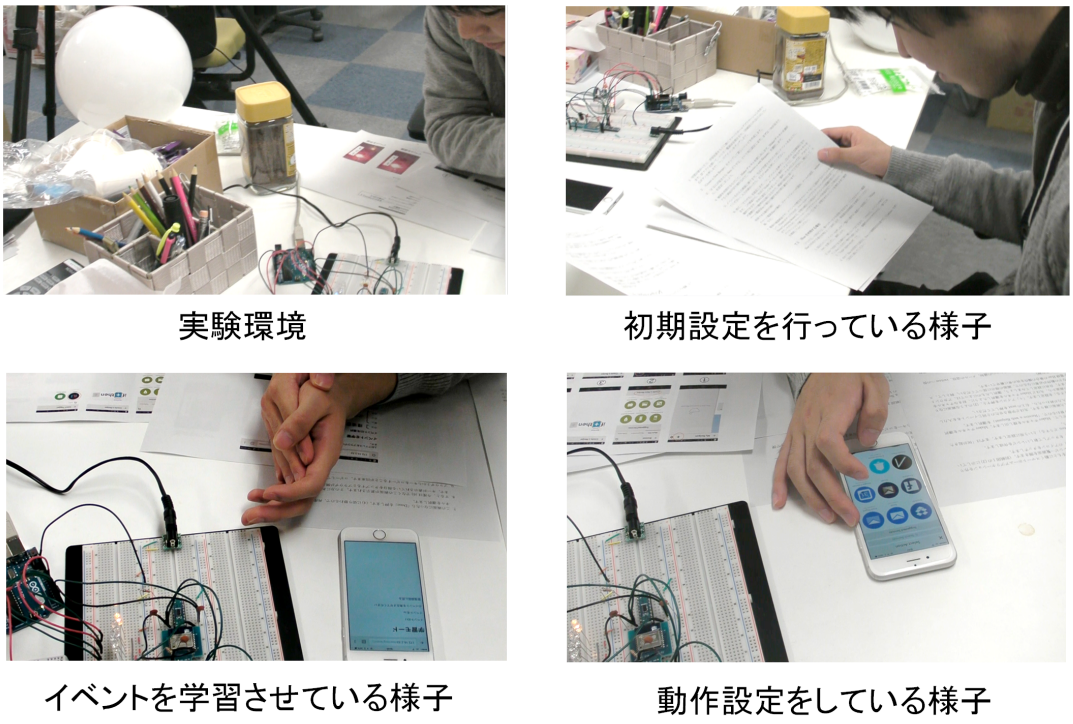

図 4.9: ユーザ B の試用評価の様子

考察と議論

システムの動作状況， 特に学習しているタイミングがわからないという意見に対しては，動作確認用の LED を見直した．実験を行った際は，LED を 4 つ使用して，学習してある 4 つのイベントのうち，どのイベントを認識したかという表示しかしていなかった．そのため，学習が行われたタイミングが把握できず，イベントを発生させた際に本当に学習しているのかが不明瞭だった．そこで，プロトタイプ実装 (2) において LED を 1 つ追加した．LED を追加したことで，Event Learning Sensor の動作状態を確認することが可能となった．詳細は，プロトタイプ実装 (2) で述べる．

音の拾える範囲については，発生する音の大きさによっても左右されるため，現状では具体的な範囲を表すことが出来ない．そのため，現状ではなるべく音の発生源の近くに設置するよう指示するに留まる．

認識してから設定した動作が行われるのに時間がかかるという点は，IFTTT が原因である可能性が高い．IFTTT への通知は，HTTP の GET リクエストを利用しているため，通常の PC のブラウザからも同じように通知を送信できる．これを利用して PC から IFTTT へイベントの認識通知を送信したものの，Event Learning Sensor でイベントを認識した場合と同程度の時間がかかった．この問題に関しては，改善を待つか他のサービスの利用を検討するほかない．

トラブルが発生し，口頭での指示が必要になったという点については，動作の安定性を高めるとともに，説明書にトラブルシューティングの項目を設けることで改善できる可能性がある．指摘された説明書の不備や改善点も含めて，試用評価 (2) に向けて説明書を更新する．

4.2.5 ノイズ調査

Event Learning Sensor のノイズ調査を行う．試用評価 (1) を行っている際に，電源の取り方によって学習・認識部が誤作動を起こし，正常に動作しなくなることがあった．基板に実装するプロトタイプ実装 (2) に向けて，どのような電源なら正常に動作するのか，また誤作動の原因がどこにあるのか探るためにノイズ調査を実施した．プロトタイプ実装 (2) では Arduino ではなく，Arduino のブートローダを書き込んだ AVR マイコンを利用するため，これを想定してブレッドボード上に回路を組んだ．ノイズ調査はエアコンの動作音程度しか音の発生していない，低騒音な研究室環境にて実施し，アンプの出力をオシロスコープで観察した．電源の種類と Wi-Fi モジュールの動作状況を変えて出力波形を観察した．実験の様子を図 4.10 に示す．

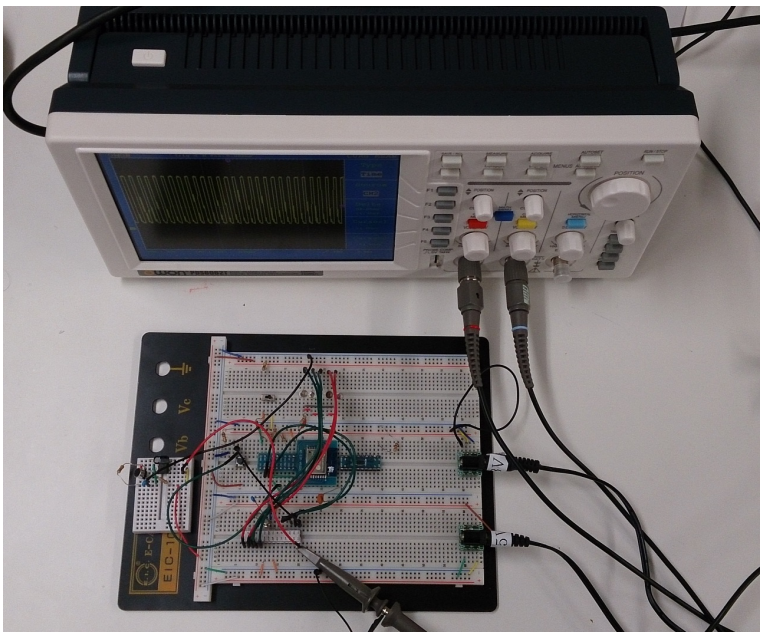

図 4.10: ノイズ調査の様子

結果

はじめに，正常に動作することが確認されている，プロトタイプ実装 (1) の波形を観察する．プロトタイプ実装 (1) では，USB 接続の Arduino から 5V を取って周辺回路を駆動させ，3.3V の AC アダプタを利用して Wi-Fi モジュールを駆動させていた．観察できた波形を図 4.11 に示す．また，正常に動作するプロトタイプ実装 (1) の状態で音を発生させたときの波形を図 4.12 に示す．

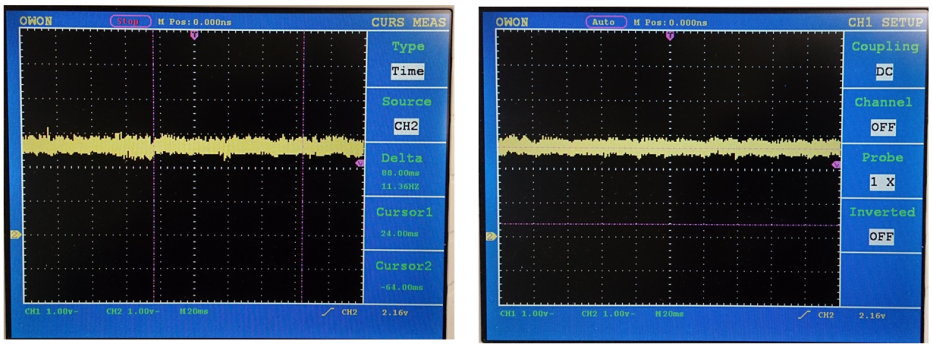

図 4.11: プロトタイプ実装 (1) の場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

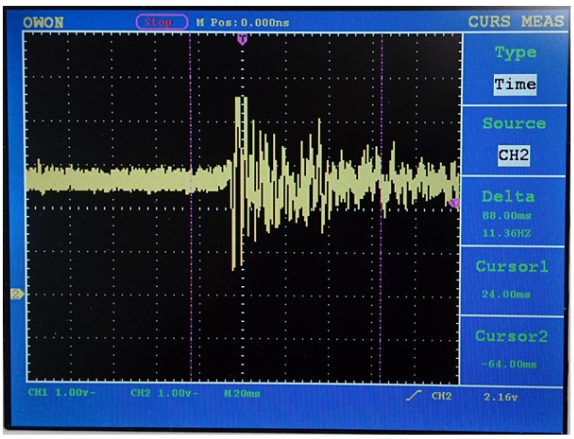

図 4.12: プロトタイプ実装 (1) で音を発生させた場合

次に，基板への実装を想定して Arduino ではなく AVR マイコンを用い，さまざまな電源で検証を行った．観察できた波形を図 4.13 から図 4.17 に示す．図 4.13 は 5V の AC アダプタから電源を取って周辺回路と AVR マイコンを駆動させ，5V の電源を DC-DC コンバータ利用して 3.3V に降圧し，Wi-Fi モジュールを駆動させたときの実験結果である．図 4.14 は 5V の AC アダプタから電源を取り，降圧にリニアレギュレータを利用したときの実験結果である．図

4.15 は 5V の AC アダプタと 3.3V の AC アダプタを利用して電源を取ったときの実験結果である．図 4.16 は単 3 電池 3 本 (4.5V) から電源を取り，降圧にリニアレギュレータを利用した時の実験結果である．図 4.17 は単 3 電池 3 本 (4.5V) と単 3 電池 2 本 (3.0V) から電源を取ったときの実験結果である．なお，DC-DC コンバータの入力電圧が 4.75V 以上だったため，電池の場合には DC-DC コンバータでの動作確認ができなかった．

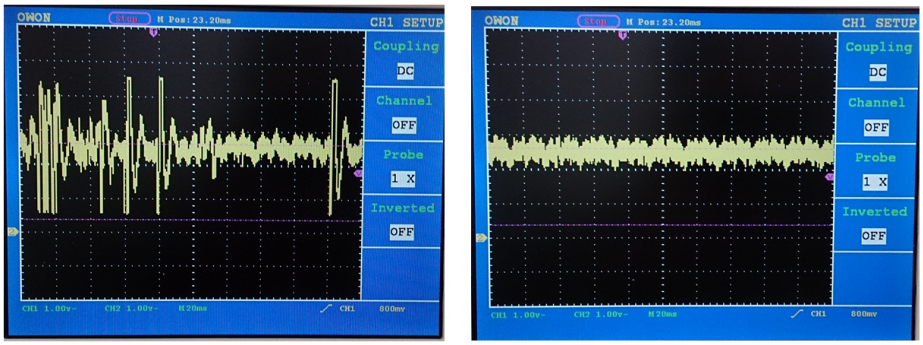

図 4.13: 5V の AC アダプタと DC-DC コンバータから電源をとった場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

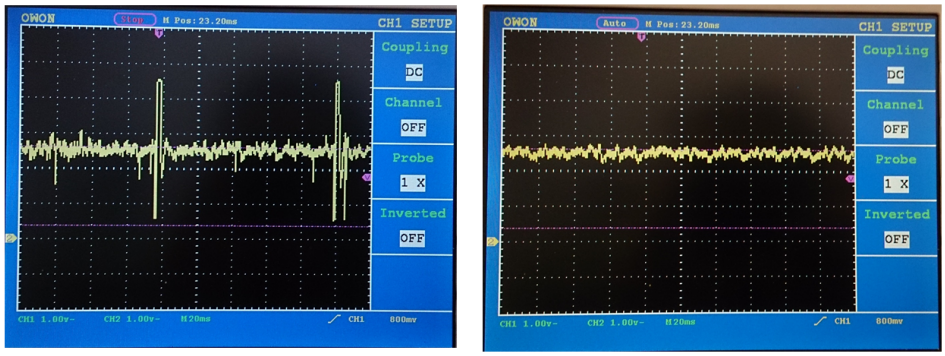

図 4.14: 5V の AC アダプタとレギュレータから電源をとった場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

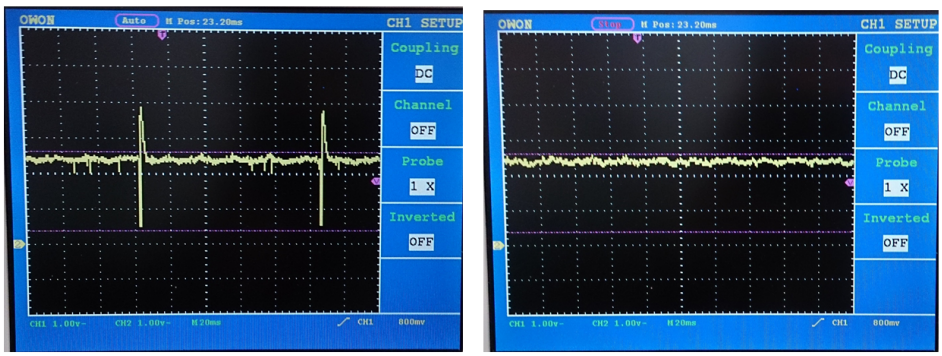

図 4.15: 5V の AC アダプタと DC-DC コンバータから電源をとった場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

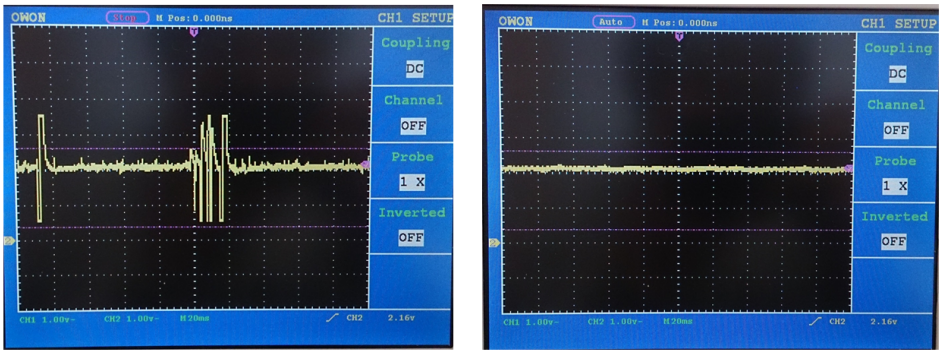

図 4.16: 単 3 電池 3 本とレギュレータから電源をとった場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

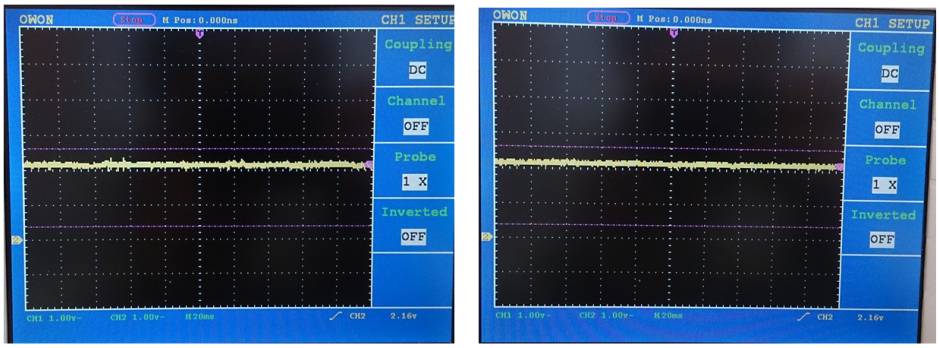

図 4.17: 単 3 電池 3 本と単 3 電池 2 本から電源をとった場合 (左)Wi-Fi/ON (右)Wi-Fi/OFF

考察と議論

この結果から，Wi-Fi モジュールの電源が ON の状態で明らかなノイズが観察できるため，ノイズの原因は Wi-Fi モジュールであると言える．しかし，図 4.17 の単 3 電池 3 本 (4.5V) と単 3 電池 2 本 (3.0V) においては，他の電源と比較して Wi-Fi モジュールからのノイズの影響が極めて小さかった．また，この実験結果からは，Wi-Fi の ON/OFF だけでなく，電源によってもノイズの発生具合が異なることがわかる．Wi-Fi/ON の状態で比較すると，急に振幅が大きくなる回数が，DC-DC コンバータを利用した図 4.13 でかなり多くなっていることがわかる．さらに，Wi-Fi/OFF の状態で比較しても図 4.13 では他と比較して振幅が大きい．そのため，DC-DC コンバータもノイズを発生させているのではないかと推測できる．

4.3　プロトタイプ実装 (2)

プロトタイプ実装 (2) ではプロトタイプ実装 (1) で得られたユーザからのフィードバックや，ノイズ調査の結果を元に，基板への実装を行う．基板に実装するため，ArduinoUNO の代わりに Arduino のブートローダを書き込んで Arduino UNO として動作するようにした Atmel 社製の AVR マイコン，ATMEGA328P-PU を利用する．電源にはノイズ調査の結果から，周辺回路と AVR マイコンを駆動させるための単 3 電池× 3 個で 4.5V の電源と，Wi-Fi モジュールを駆動させるための単 3 電池× 2 個で 3.0V の電源を利用する．プロトタイプ実装 (1) のときには Arduino と Wi-Fi モジュールで別々に合ったリセットボタンを一つにまとめ，ユーザにもわかりやすいように「RST」の表示を行った．

試用評価 (1) の結果を受けて追加した LED は，学習・認識を待機している際に点灯し続けるようにした．そして，大きな音を検知して学習・認識を行う際には一旦消灯する．認識の場合には従来通り認識した 0-3 のイベント ID に対応する LED を点灯させる．なお，学習したイベントとして認識できなかった場合はどの LED も点灯しない．学習の場合には，追加した LED と学習を指示した 0-3 のイベント ID に対応する LED が同時点灯する．LED の点灯状態を図 4.18 に示す．

図 4.18: LED の点灯による状態表示

作成したプロトタイプ実装 (2) の回路図を図 4.19 に，基板上に実装する際の配線図を 4.20 に，実際に作成したものを図 4.21 に示す.

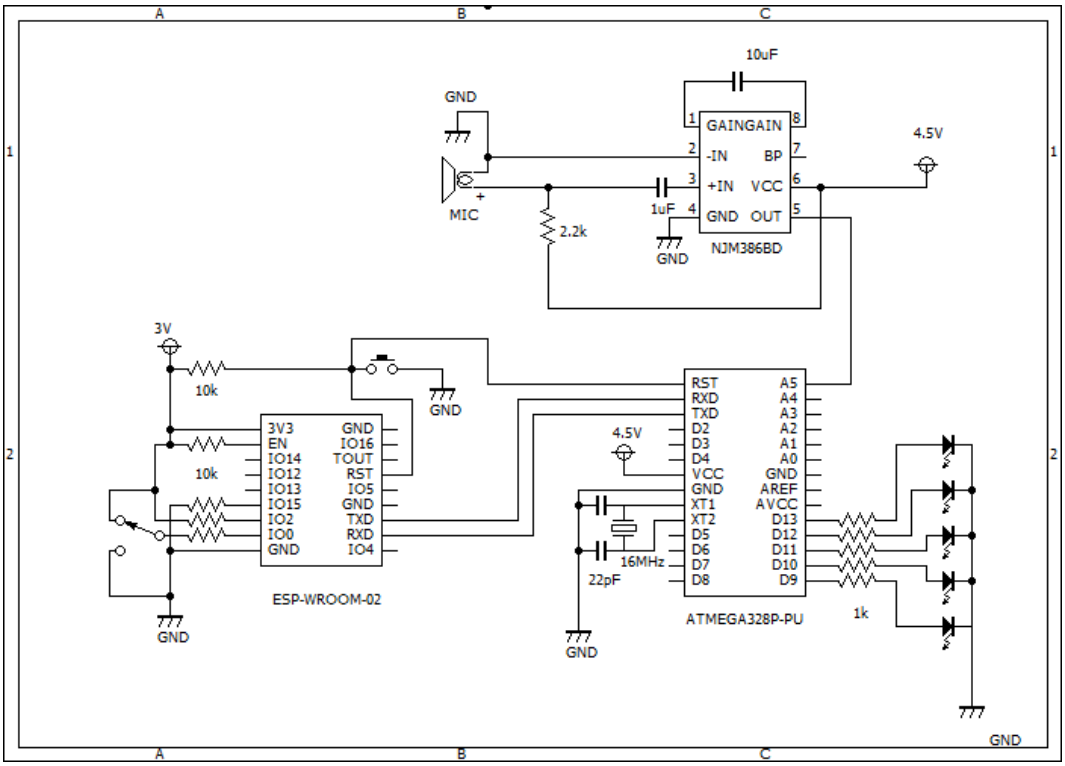

図 4.19: プロトタイプ実装 (2) の回路図

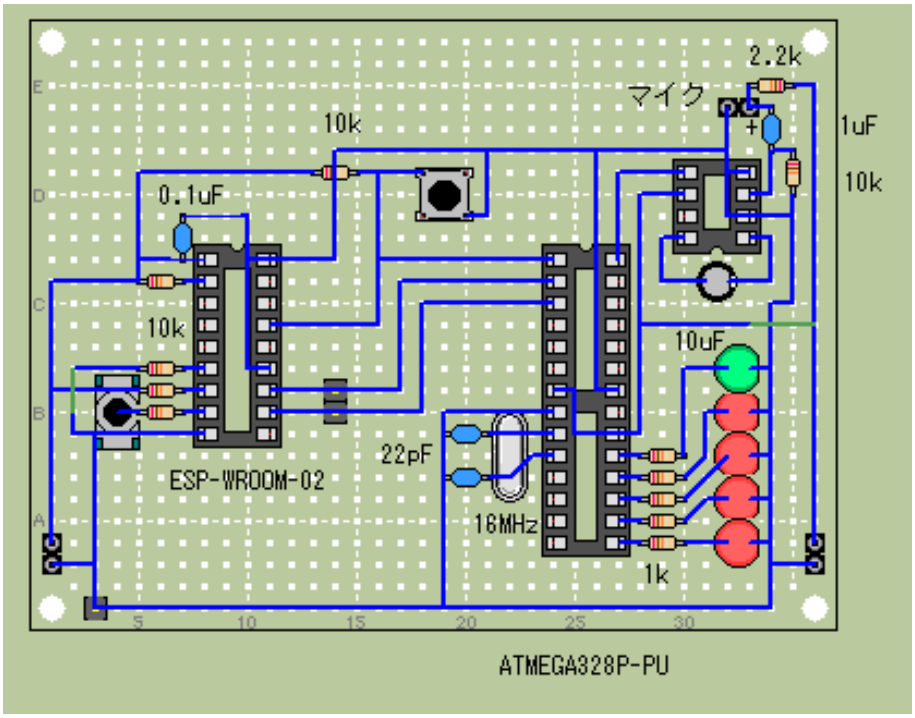

図 4.20: プロトタイプ実装 (2) の配線図

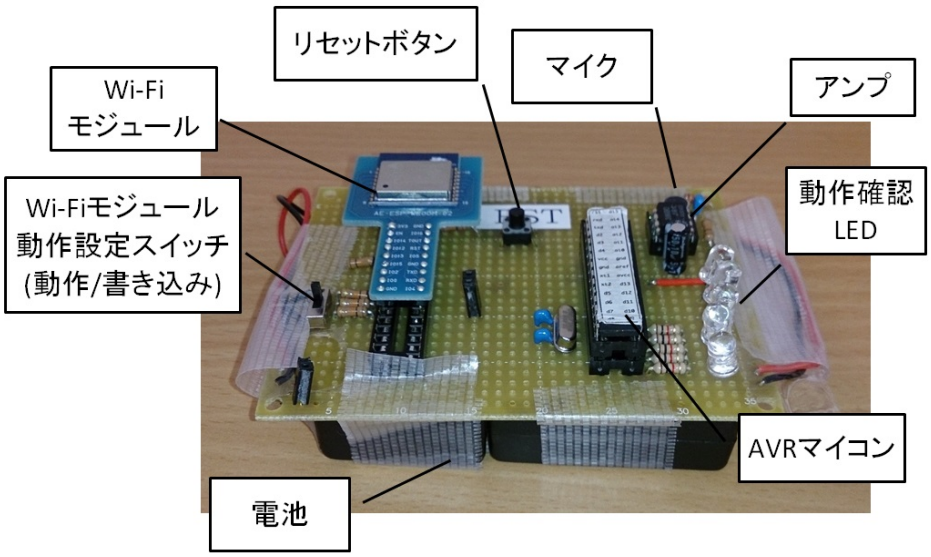

図 4.21: プロトタイプ実装 (2) の実物

4.3.1 認識率の調査

プロトタイプ実装 (2) を対象に，イベント認識率の調査を行った．エアコンのファンの音程度の雑音しか発生していない低騒音な研究室で実験を実施した．実験環境を図 4.22 に示す．

この実験では，4 つのイベントを学習させたあと，学習させたイベント 4 つと学習させていないイベント 1 つの合計 5 つのイベントを発生させ，認識結果を Event Learning Sensor の LED で確認した．学習させたイベントは，

- 電子レンジの調理が完了した (電子レンジのブザーが鳴ったタイミングで学習・認識)
- 冷蔵室ドアの開閉 (冷蔵室のドアを閉めたタイミングで学習・認識)
- 冷凍室ドアの開閉 (冷凍室の引き出しを閉めたタイミングで学習・認識)
- 机をノックした (机を手の甲で叩いたタイミングで学習・認識)

の 4 つである．また，学習していないイベントとして，電子レンジのドアを開けた (電子レンジのドアの開けたタイミングで学習・認識) を発生させた．冷蔵室・冷凍室は，開ける時より閉めた時のほうが大きな音が発生する．学習・認識のタイミングは，大きな音の発生をトリガとしているため，閉めたタイミングで学習・認識が行われた．Event Learning Sensor と音の発生源との距離は，電子レンジ・冷蔵庫とは約 30cm，机のノックする場所とは約 50cm 離れている．

結果

学習させる 4 つのイベント全てを学習させたあと，認識させる 5 つのイベントをそれぞれ 10 回ずつ発生させた．実験の結果を表 4.2 に示す．

表 4.2: 認識率調査の結果 [回]

認識したイベント	発生させたイベント				
	レンジブザー	冷蔵室	冷凍室	机ノック	レンジドア
レンジブザー	9	0	0	0	0
冷蔵室	0	7	1	0	2
冷凍室	0	0	1	0	4
机ノック	0	0	1	10	0
学習していないイベント	1	3	7	0	4
認識率	0.9	0.7	0.1	1	0.4

全体の認識率は，0.62 であった．この結果をみると,「冷凍室ドアの開閉」というイベントの認識率がかなり低いことがわかる．

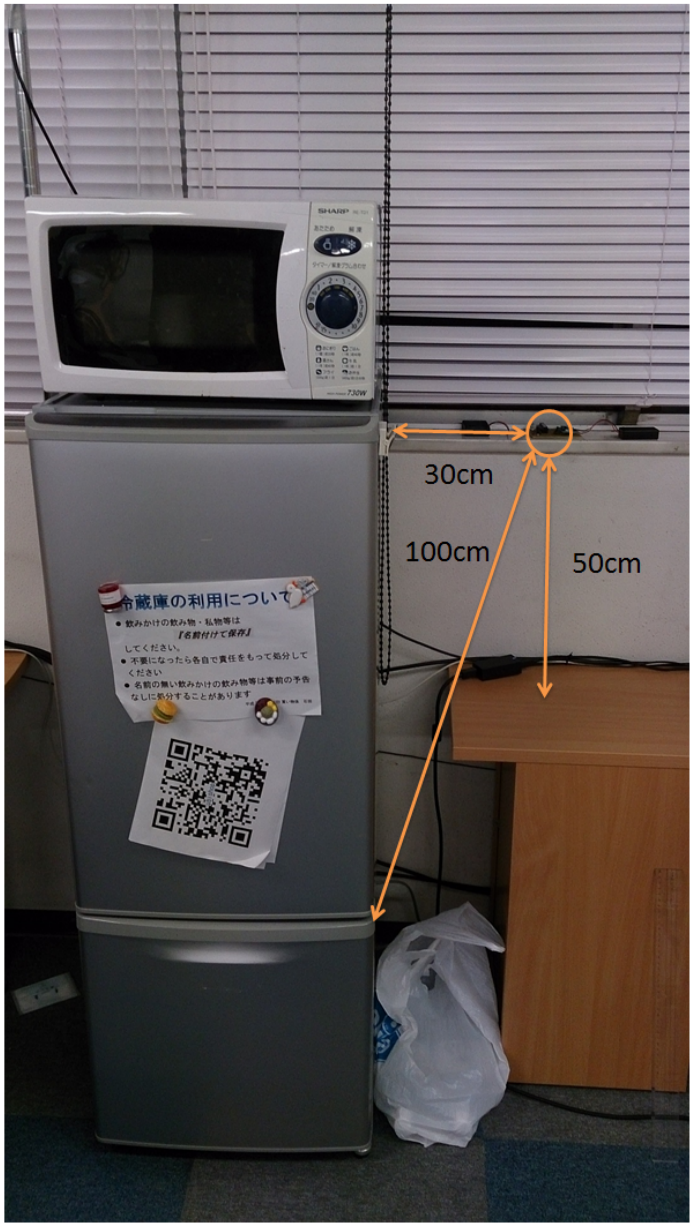

図 4.22: 認識率調査実験の実験環境

追実験

「冷凍室ドアの開閉」の認識率がなぜここまで低くなったのか，原因を解明するために追実験を行う．追実験では学習・認識に用いている正規化した振幅スペクトルと，イベント認識時の 2 乗誤差の和を取得した．まずは冷凍室のドアを 3 回開閉し，それぞれの正規化した振幅スペクトルを比較し，どれだけ似た音が発生しているのか観察した．取得した正規化振幅スペクトルを図 4.23 に示す．

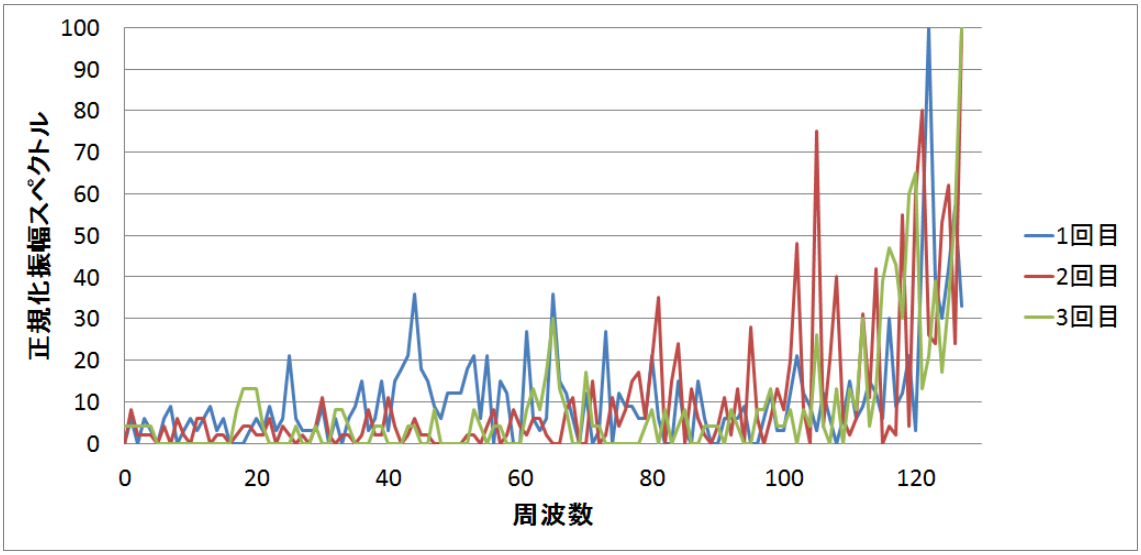

図 4.23: 冷凍室ドア開閉時の正規化振幅スペクトル 3 回分

次に，実際の認識時に学習・認識部がそれぞれの音がどれだけ近いか判断するための 2 乗誤差の和を比較する．各イベントを学習させた後，それぞれのイベントを発生させたときに計算された 2 乗誤差の和を表 4.3 に示す．今回，特に注目している冷凍室ドアの開閉に関しては，正常に認識された場合の 2 乗誤差の和と，学習していないイベントと判断された場合の 2 乗誤差の和をのせる．認識したイベントは太字で示す．なお，学習していないイベントと判断されるのは，2 乗誤差の和の最小値が 30000 を超えた場合である．

表 4.3: 2 乗誤差の和

学習してあるイベント	発生させたイベント				
	レンジブザー	冷蔵室	冷凍室	机ノック	冷凍室 (誤認識)
レンジブザー	**19880**	57254	50190	41208	49178
冷蔵室	51282	**24758**	45722	32377	58766
冷凍室	44198	39932	**26950**	32603	37894
机ノック	26933	49993	44395	**24730**	48211

考察と議論

図 4.23 を見てみると，3 回とも似たようなグラフを描いているとは言い難い．認識の大きな特徴となる振幅スペクトルの最大値も 2 回目と 3 回目は同じ周波数だが，1 回目は異なる周波数にある．このことから，同じイベントを発生させても同じような振幅スペクトルが観察できるわけではないということがわかる．同じイベントで同じような振幅スペクトルを観察することが出来ないという点が，冷凍室ドアの開閉の認識率が非常に低くなった原因の 1 つといえる．

表 4.3 の冷凍室ドアの開閉が認識できなかった場合の数値をみてみると，しきい値の 30000 は超えて学習していないイベントと判断はされているものの，冷凍室ドアの開閉が最も小さくなっている．そのため，学習していないイベントと判断するためのしきい値の調整によって，認識率の改善ができる可能性もある．

実験前には，電子音である電子レンジのブザーが最も高い認識率となると予想していた．しかし，認識率調査の結果，10 回中 1 回だけ，学習していないイベントとして認識された．表 4.3 を見てみても，電子レンジのブザーの誤差が最も小さくなっている．このことから，電子レンジのブザーは他のイベントと最も差別化しやすいと言える．それにもかかわらず，学習していないイベントとして認識されたのは，何らかのノイズの影響も考えられ，より詳細な実験を行い，検証する必要がある．

4.3.2 電池寿命調査

今回は，Event Learning Sensor を電池駆動としたため，電池がどのくらい持つのかという評価を行った．学習・認識部のプログラムを少し改変し，通常の動作とは別に 10 分毎にタイマ割り込みを発生させ，Wi-Fi モジュールへダミーイベントの認識通知を送信するようにした．Wi-Fi モジュールは通常の動作として，この通知を受け取ると IFTTT への通知を行う．今回は，この IFTTT への通知をスマートフォンで受信し，どのくらいの間通知が送られてくるかを観察した．

実験に使用した電池は，新品の Panasonic EVOLTA 電池単 3 形である．これを計 5 本セットして実験を開始した．実験開始後は，通常の利用と同様に初期設定を行い，イベントを一つ学習させたあと，ダミーイベントの通知をスマートフォンで受信するように設定を行った．

実験の結果，約 35 時間後の通知を最後に通知が停止した．実際の環境では，通知を行う回数が異なるため一概に言えないものの，電池では実用度が低いと考えられる結果となった．実用性を高めるためには，リチウムイオン電池などのエネルギー密度の高い電池を利用する，省電力性能を向上させるといった対策が必要である．またはノイズ問題を解決し，AC アダプタを利用して常に電力供給を受けるという対策も考えられる．

4.4　プロトタイプ実装 (2) の使用例

Event Learning Sensor のプロトタイプ実装 (2) を利用して，実際にスマートホームアプリケーションを作成する手順を示す．利用イメージで示した，ドアを開けたら部屋の照明が点灯するというアプリケーションの作成を例にとって説明する．ここでは，部屋の照明としてスマート電球 Hue を利用する．

4.4.1　事前準備

事前準備としてスマートフォンに IFTTT のアプリをダウンロードしてアカウントを作成し，連携に必要なキーを取得する．操作の流れを図 4.24 に示す．

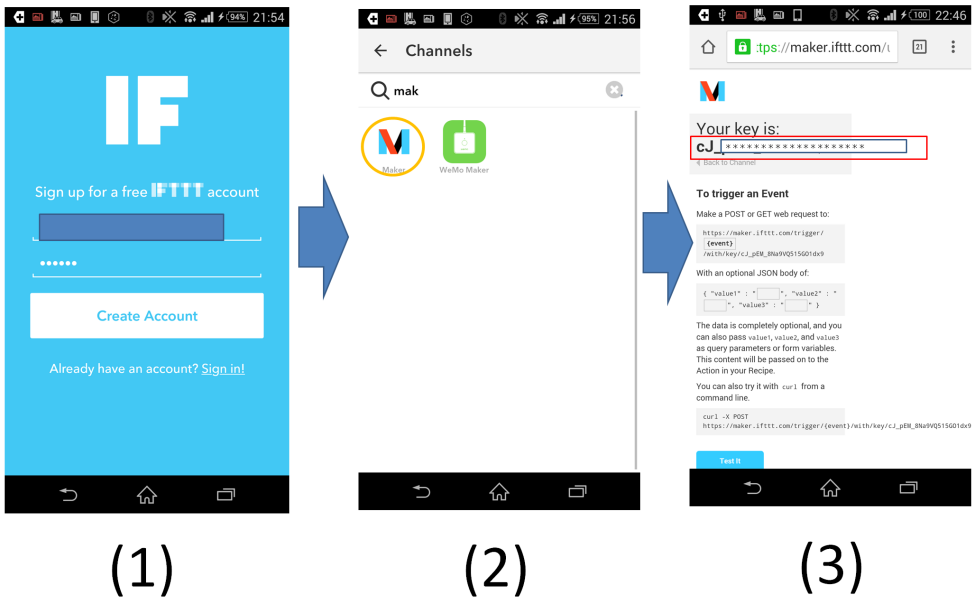

図 4.24: 事前準備の流れ

1. スマートフォンに IFTTT アプリをダウンロードして，アプリを開くと会員登録画面になる．この画面でメールアドレスを入力し，パスワードを決めてアカウントを作成する．(図 4.24(1))

2. アカウントの作成が終わったら，IFTTT アプリの設定画面からチャネル一覧を開き，Maker チャネルを選択する．(図 4.24(2))

3. 先に進んでいくと IFTTT との連携に必要なキーが表示されるので，これをコピーする．(図 4.24(3))

4.4.2　設置

どんなイベントを学習させるかを考え，Event Learning Sensor を設置する．今回は，ドアを開けるというイベントを認識させるため，ドア付近に設置した．設置の様子を図 4.25 に示す.

図 4.25: 設置の様子

4.4.3　初期設定

Event Learning Sensor のアクセスポイントに接続し，初期設定を行う．操作の流れを図 4.26 に示す.

1. スマートフォンの Wi-Fi 設定から Event Learning Sensor のアクセスポイントに接続する．(図 4.26(1))

2. スマートフォンのブラウザを開き，アドレスバーに「192.168.4.1」と入力し，初期設定画面にアクセスする．初期設定画面で，リストから接続したい Wi-Fi の SSID（アクセスポイント名）を選択し，パスワードを入力する．さらに，事前準備で取得した IFTTT のキーを入力する．(図 4.26(2))

3. 設定情報を送信すると，設定完了の画面が表示される．設定完了画面の指示に従って Event Learning Sensor のリセットボタンを押し，再起動させる．リセットボタンを押す様子を 4.27 に示す．再起動後は，スマートフォンと Event Learning Sensor の接続が切れるため，再度 (1) のように Event Learning Sensor に接続する．(図 4.26(3))

図 4.26: 初期設定の操作の流れ

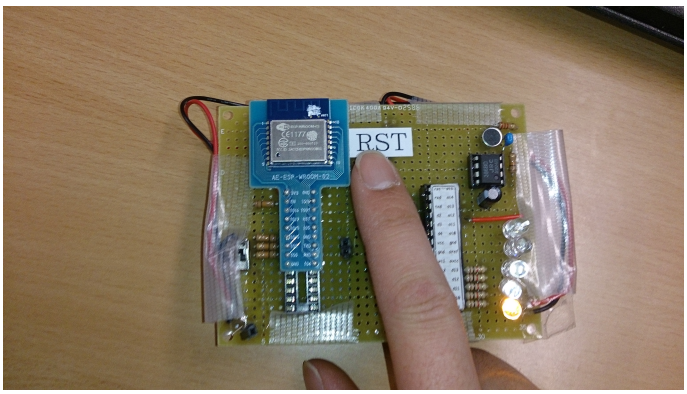

図 4.27: リセットボタンを押す様子

4.4.4　イベントの学習

Event Learning Sensor の設定画面を表示させ，学習させる．学習の流れを図 4.28 に示す．

図 4.28: 学習の流れ

1. スマートフォンのブラウザから，初期設定と同じように「192.168.4.1」に接続する．すると，今度は Event Learning Sensor の設定画面が表示される．Event Learning Sensor には，合計 4 つのイベントを学習させることができる．各イベントにはイベント ID が割り振られているが，初期状態ではどのイベント ID も空である．今回は，イベント ID.1 にドアが開いたというイベントを学習させる．ブラウザの画面でイベント ID.1 を選択し，イベント名の欄に「DoorOpen」と入力し，実行ボタンを押す．(図 4.28(1))

2. すると，画面の表示が切り替わり，Event Learning Sensor が学習待機状態になる．この画面になったら，実際にイベントを発生させる．実際にイベントを発生させている様子を図 4.29 に示す．学習待機状態のときにしきい値を超える大きな音が発生したら，Event Learning Sensor は学習処理を行う．今回は，イベント ID.1 に学習を行ったので，上から 2 番目の赤色 LED と一番下のオレンジ色の LED が点灯する．LED の点灯状態を図 4.30 に示す．(図 4.28(2))

図 4.29: 学習の様子

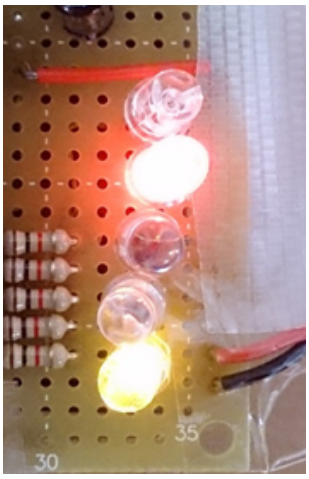

図 4.30: イベント ID.1 に学習していることを示す Event Learning Sensor の LED 表示

4.4.5 IFTTT での動作設定

認識したイベントをトリガとしてどんな動作をするか，IFTTT のアプリから設定する．IFTTT のアプリから動作の設定をする際の流れを図 4.31 に示す.

1. スマートフォンで IFTTT のアプリを開き，レシピの新規作成画面を開く．まずは，トリガとなる if の部分を設定する．(図 4.31(1))
2. さまざまなチャネルの中から，Maker チャネルを探して選択する．すると，イベント名の入力を求められるので，学習時に入力したイベント名「DoorOpen」を入力する．(図 4.31(2))
3. if の部分の設定が終わったら何をするかという then の部分の設定を行う．(図 4.31(3))
4. then の部分として，スマート電球 Hue の明かりがつくように設定する．さまざまなチャネルの中から Hue を選択する．すると，Hue にどんな動作をさせるかの一覧が出てくる．今回は，明かりがつくだけでなく，ON/OFF の切り替えができるように,「Toggle lights on/off」を選択した．(図 4.31(4))
5. 最後に，作成したイベントの確認画面が表示される．確認して問題がなければ，Finish を押して動作設定を完了させる．(図 4.31(5))

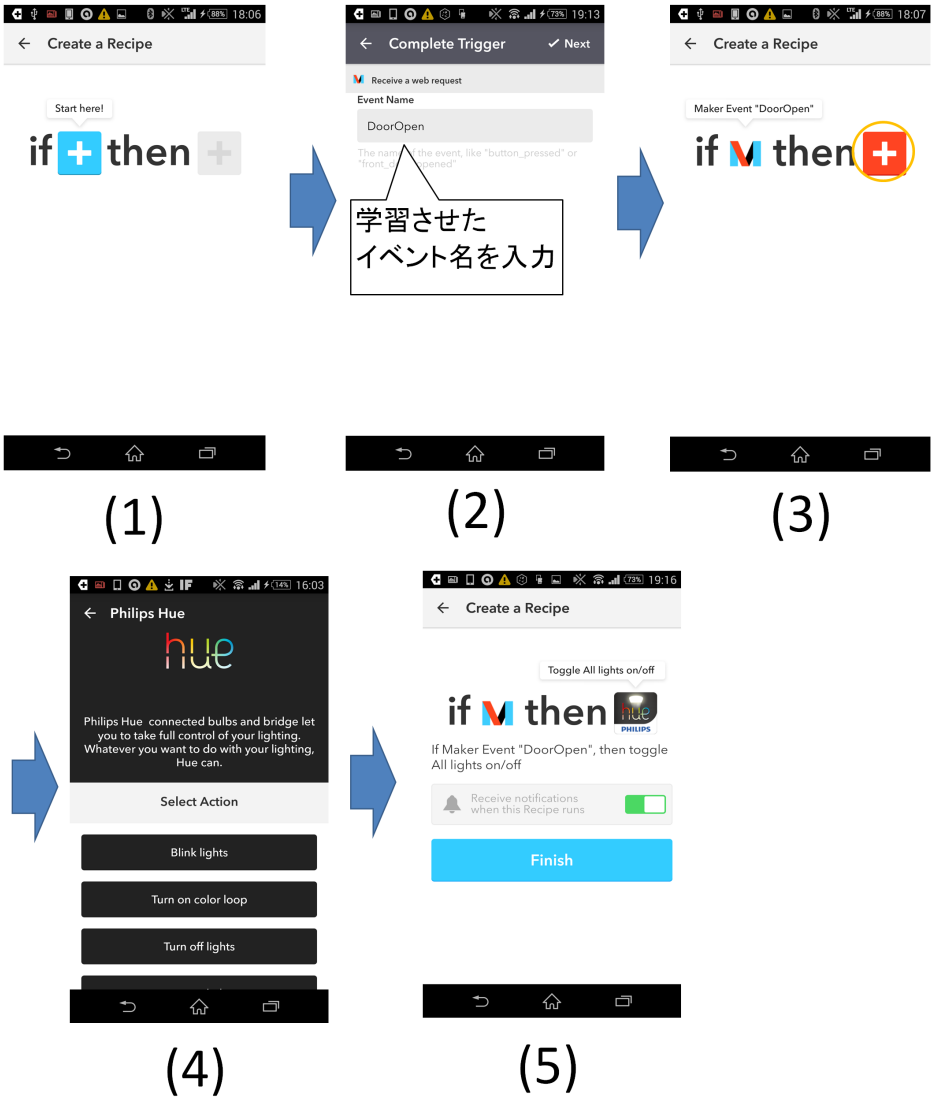

図 4.31: 動作設定の流れ

4.4.6　作成したスマートホームアプリケーションの利用

ドアを開けると Event Learning Sensor はそのイベントを認識し，IFTTT に通知を送信する．IFTTT は Hue に電源の ON/OFF を切り替えるように命令を行う．すると，Hue が消灯している状態であれば点灯するようになる．実際に，ドアを開けたら Hue が点灯した様子を図 4.32 に示す.

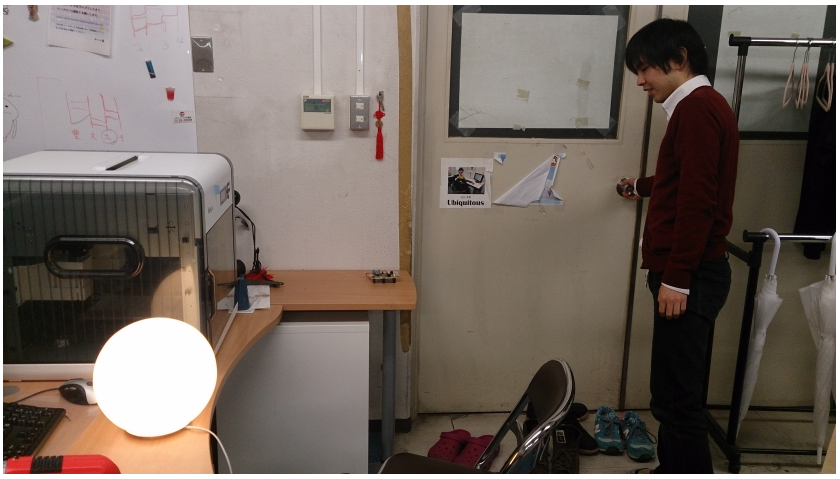

図 4.32: 実際にドアを開けて Hue を点灯させる様子

第5章　試用評価(2)

プロトタイプ実装(2)の試用評価を行い，プロトタイプ実装(1)で指摘された点が改善されているか，また他に改善すべき点がないかなどを調査する．プロトタイプ実装(1)での試用評価(1)と同様に，初期設定からアプリケーションの動作を確認するところまでの一連の操作を被験者に行ってもらい，その時の様子を観察し，インタビューを行った．

5.1　被験者

22歳から24歳までの情報系学生3名(試用評価(1)のユーザA，Bに加え，ユーザCとする)

5.2　タスク

試用評価(1)と同様に，研究室でプロトタイプ実装(2)のEvent Learning Sensorと被験者自身のスマートフォンを利用して試用評価を行った．また，連携するスマートホーム製品として，スマート電球Philips Hueを用意した．被験者は，Event Learning Sensorの取扱説明書を見ながら，事前準備から作成したアプリケーションの動作を確認するところまでを実施する．アプリケーションの動作確認が終わったら，使用感等をインタビューする．試用評価(2)で利用したEvent Learning Sensorの取扱説明書を付録A,Bとして添付する．実際に利用することを想定し，被験者にはなるべく説明書のみを参考にして作業するように指示をした．ただし，被験者が途中で困ってしまった場合には口頭で説明を行った．なお，ユーザA，Bは事前準備であるIFTTTのアカウント取得は試用評価(1)で既に行っているため，今回は実施しない．その代わり，インタビューでプロトタイプ実装(1)との使用感の変化等も聞いた．

5.3　評価手順

被験者の実験の流れは以下の通りである．試用評価(1)では，学習を行ってからIFTTTの動作設定が完了するまでの間に，正常に認識されることを確認するステップがなかったため今回はそれを追加した．

1. 実験内容の説明を受け，取扱説明書を受け取る．
2. 事前準備としてIFTTTのアカウントを取得する．(ユーザA，Bは行わない)

3. Event Learning Sensor の初期設定を行う．
4. Event Learning Sensor の設定画面を表示させる．
5. トリガとしたいイベントを決め，学習させる．
6. 正常に学習されているか，イベントを発生させて確認する．正常に学習されていなかった場合は再学習を行う．
7. 認識したイベントをトリガとしてどんな動作をするか，IFTTT のアプリから設定する．
8. 設定したとおりに動くことを確認する．
9. 感想や改善してほしい点があれば伝える．試用評価 (1) を行っていた場合は，変わった点も伝える．

5.4 結果

試用してもらった結果，被験者が実際に作成したアプリケーションを紹介する．ユーザ A は「研究室のドアが開いたらメールで通知を送る」というアプリケーションを作成した．ユーザ B は「机を叩いたらスマート電球の色が変化する」というアプリケーションを作成した．ユーザ C は「冷蔵庫のドアを開いたらメールで通知を送る」というアプリケーションを作成したユーザ C の試用評価の様子を図 5.1 に示す．

今回で 2 度目の利用となるユーザ A，B のタスクにかかった時間は，初期設定からアプリケーションの動作確認ができるまでで共に約 10 分だった．今回初めて Event Learning Sensor を利用するユーザ C のタスクにかかった時間は，事前準備からアプリケーションの動作確認ができるまでで約 27 分だった．

被験者の様子を観察した結果， プロトタイプ実装 (2) でも問題なく利用イメージを一通りこなせることが確認できた．今回加えた正常に学習されているかをチェックするプロセスにより，ユーザ A，B は再学習を行った．正常に学習されているかをチェックするプロセスを加えたことについて，ユーザ A はその場で認識結果が確認できてよかったと話した．また，ユーザ B はページが切り替わらないという問題に遭遇した際，自分でリセットボタンを押した後にスマートフォンと Event Learning Sensor との接続を再確認することで，問題の自己解決を行った．

インタビューにおいてユーザ A，B に前回との違いを尋ねた結果，どちらも LED の追加により動作状況がよく分かるようになり，便利になったと答えた．特に，ユーザ A は学習のタイミングが確認できるようになったことを評価した．また，ユーザ A は 2 回目なのでスムーズにできた，特に IFTTT を用いたスマートホームアプリケーションの作成に戸惑うことが無くなったと答えた．今回初めて利用したユーザ C も，初期設定が煩わしいが 2 回目以降の利用は初期設定も不要な上，操作にも慣れると思うと答えた．

実験環境

初期設定を行っている様子

イベントを学習させている様子

作成したアプリケーションが動作し、
メールが届いた

図 5.1: ユーザ C の試用評価の様子

5.5 考察と議論

ユーザ A，B のタスク完了にかかる時間は 2 回目ということもあり，かなり短縮された．試用評価 (1) では，初期設定からアプリケーションの動作確認までユーザ A は約 22 分，ユーザ B は約 18 分であった．ユーザ A が「2 回目なのでスムーズに出来た」，ユーザ C が「2 回目以降は操作になれるだろう」と答えているように，何度か利用すればより短時間でスマートホームアプリケーションの作成ができるようになることを示唆している．

学習をチェックするプロセスを追加したことについては，その場で認識結果を確認し適切に学習されているかどうかを確認できる，として好評を得ることが出来た．ただ，再学習を行う必要が出たという点は学習・認識がうまく行えていないということを示唆している．そのため，今後は学習・認識手法の改善を行っていく必要があるといえる．リセットボタンを明示したことについては，実際にユーザ B がリセットボタンを利用して，問題の自己解決を行い，タスクを続行した．試用評価 (1) では介入することでトラブルを解決していたが，今回は自己解決ができたことから専門家のアドバイス無しに利用できるという目標に近づけたといえる．

前回との比較については，指摘した点が改善されたとの評価を得た．特に，LED の追加に関しては好評を得ることが出来た．また，ユーザ C のタスク完了にかかる時間は約 27 分と，試用評価 (1) のユーザ A のタスク完了時間約 40 分，B のタスク完了時間約 35 分よりも早い結果となった．これらの結果から，プロトタイプ実装 (1) よりもプロトタイプ実装 (2) の方が有用なシステムになったと言える．

第6章　結論

本研究では，スマートホームユーザ向けのイベント学習・認識システムである Event Learning Sensor を開発した．プロトタイプ実装 (1) での試用評価 (1) で，開発した Event Learning Sensor を用いて実際にイベントを学習・認識させ，スマートホームアプリケーションの作成ができるようになったことを確認した．試用評価 (1) でユーザから得られたフィードバックに基いてシステムの改善を行い，プロトタイプ実装 (2) を作成した．プロトタイプ実装 (2) での試用評価 (2) によって，プロトタイプ実装 (1) でユーザに指摘された点が改善されたことを確認した．今回は，Event Learning Sensor の学習・認識部分を簡易的に実装した．プロトタイプ実装 (2) で行った認識率調査の結果，認識率は 0.62 であった．また，試用評価において学習・認識が正常に行えていないケースも確認された．

今後の課題として，第一に認識率を向上することが挙げられる．認識率を向上させるため SVM などのより高度な手法で学習・認識部を再実装する必要がある．その上で，改めて認識率の測定を行い，性能が向上したのかを確認する必要がある．新たに実装した学習・認識手法において，ユーザの使用感がどのように変化するか，試用評価を改めて行う必要がある．また，今回は情報系学生に対する試用評価を行ったため，今後は情報分野を専門としていない人を対象とした評価実験を行い，本システムの有用性を確かめる必要がある．システムの今後の展望として，今回はイベントの認識に音を利用したが音以外のセンサも併用して，より多くのイベントを学習・認識できるようにしたい．

謝辞

本論文を執筆するにあたり，指導教員である高橋伸准教授，田中二郎教授をはじめ，志築文太郎准教授にはゼミやミーティングを通して大変貴重なご意見，アドバイスをいただきました．深く御礼申し上げます．また，インタラクティブプログラミング研究室の皆様にはゼミや日常生活の中で数々のご意見をいただきました．特に，ユビキタスチームの皆様にはゼミ以外にも研究生活・大学生活全体にわたって数多くのご意見やご指摘をいただきました．この場を借りて厚く御礼申し上げます．そして最後に，大学生活を送る中，経済面や精神面にわたっても支えてくれた両親，そして大学生活を共に過ごし様々な面でお世話になった全ての友人に心より感謝いたします．

参考文献

[1] Treffyn Lynch Koreshoff, Toni Robertson, and Tuck Wah Leong. Internet of things: a review of literature and products. In *Proceedings of the 25th Australian Computer-Human Interaction Conference: Augmentation, Application, Innovation, Collaboration (OzCHI '13)*, pp. 335-344, 2013.

[2] Sarah Mennicken, Jo Vermeulen, and Elaine M. Huang. From today's augmented houses to tomorrow's smart homes: new directions for home automation research. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*, pp. 105-115, 2014.

[3] MESH `http://meshprj.com/jp/`

[4] A.J. Bernheim Brush, Bongshin Lee, Ratul Mahajan, Sharad Agarwal, Stefan Saroiu, and Colin Dixon. Home automation in the wild: challenges and opportunities. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*, pp. 2115-2124, 2011.

[5] Jong-bum Woo, Youn-kyung Lim. User Experience in Do-It-Yourself-Style Smart Homes. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*, pp. 779-790, 2015.

[6] Blase Ur, Elyse McManus, Melwyn Pak Yong Ho, Michael L. Littman Practical Trigger-Action Programming in the Smart Home. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*, pp. 803-812, 2014.

[7] 大内 一成, 土井 美和子. 携帯電話搭載センサによるリアルタイム生活行動認識システム. 情報処理学会論文誌, Vol.53, No.7, pp. 1675 - 1686. 2011.

[8] Peter Hevesi, Sebastian Wille, Gerald Pirkl, Norbert Wehn, and Paul Lukowicz. Monitoring household activities and user location with a cheap, unobtrusive thermal sensor array. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*, pp. 141-145, 2014.

[9] 根岸 佑也, 河口 信夫. 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. 情報処理学会論文誌, Vol.50, No.4, pp. 1272-1286, 2009.

[10] Michael O. Jewell, Enrico Costanza, and Jacob Kittley-Davies. Connecting the things to the internet: an evaluation of four configuration strategies for wi-fi devices with minimal user interfaces. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*, pp. 767-778, 2015.

[11] IFTTT `https://ifttt.com/`

[12] Philips Hue `http://www2.meethue.com/ja-jp/`

付 録A　Event Learning Sensor の説明書

試用評価 (2) で利用した Event Learning Sensor の説明書を示す.

A.1　はじめに

Event Learning Sensor は，様々なイベントの音を学習・認識して，スマートホームアプリケーションの作成に利用することを目的としたデバイスです．Event Learning Sensor を利用すると例えば，ドアが開く音を認識して部屋の明かりを点けるといったようなスマートホームアプリケーションを手軽に作成することができるようになります．Event Learning Sensor の外観と各部の説明を別紙図 A.1 に示します.

この手順書では，Event Learning Sensor を利用して，実際にアプリケーションを作成するために必要な手順をスクリーンショットを交えながら詳細に解説します．スクリーンショットは付録の別紙を参照してください．スクリーンショットなどは Android 端末を想定していますが，システム自体は iOS デバイスでも利用可能です.

もし，困ったことがあったら最後にトラブルシューティングのページがありますので確認してみてください.

A.2　事前準備

A.2.1　IFTTT アカウントの取得

Event Learning Sensor を利用して，スマートホームアプリケーションの作成を行う際に，IFTTT というサービスを利用します．IFTTT は，さまざまな web サービス同士を連携させることができるサービスです．IFTTT では，作成したアプリケーション（動作設定）を「レシピ」，連携させるサービスをチャネルと呼びます．まず，IFTTT を利用するために，IF アプリと IFTTT アカウントの取得が必要です．アプリストアからアプリを入手したら別紙図 A.2 のような手順で IFTTT の設定を行います.

1. メールアドレスを入力し，パスワードを決めてアカウントを取得します.
2. IFTTT のホーム画面に遷移するので，右上のボタンを押します.

3. ここは，IFTTT のレシピ一覧画面です．さらに右上の歯車のボタンを押します.
4. 設定画面で「Channels」を選択します.
5. たくさんのチャネルが表示される中から,「Maker」チャネルを選択します（上の検索バーを利用できます）.
6. 「Connect」を押して自動で画面が切り替わるのを待ちます.
7. この画面になったら「Done」を押します．(4) に切り替わるので，再度「Maker」チャネルを選択します.
8. すると，今度は (6) でなくこの画面が表示されます．下の方にある，キーを利用します．キーが表示されている部分をタップするとブラウザが開きます.
9. ブラウザ上からキーをコピーすることが出来ます．コピーしておくと初期設定で便利です.

A.3　初期設定

Event Learning Sensor の初期設定を行い，学習できる状態にします．画面の表示例は別紙図 A.3 を参照してください.

1. Event Learning Sensor の電源を入れると，アクセスポイントが立ち上がります．スマートフォンの Wi-Fi 設定から，Event Learning Sensor を選択して接続してください.
2. Event Learning Sensor に接続できたら，Web ブラウザを開き，アドレスバーに「192.168.4.1」と入力してください．すると，初期設定画面が表示されます.「設定画面へ」を押してください.
3. 表示が切り替わったら，リストから Wi-Fi の SSID（アクセスポイント名）を選択し，パスワードを入力します．さらに，事前準備で取得した IFTTT のキーを入力します．コピーしたものを貼り付けると楽です.
4. 前の画面で設定情報を送信すると初期設定完了です．Event Learning Sensor のリセットボタン（RST と書いてあるボタン）を押して Event Learning Sensor を再起動させてください.

A.4　学習画面の表示

学習画面は，初期設定と同じように，Event Learning Sensor のアクセスポイントに直接接続して表示することもできますが，先ほど設定した家のアクセスポイント経由でも表示することが出来ます．Event Learning Sensor のアクセスポイントと直接接続していると，インターネット接続が出来ませんが，いつも使っている家のアクセスポイント経由でならイ

ンターネット接続も，Event Learning Sensor の学習画面の表示もできます．いつものアクセスポイントからの学習画面の表示方法を確認します．画面の表示例は，別紙図 A.4 を参照してください．

1. 初期設定と同様に，ブラウザで「192.168.4.1」にアクセスします．この際，初期設定終了時に Event Learning Sensor が再起動しているので，Wi-Fi 設定で Event Learning Sensor につながっているか確認してください．
   この画面に表示されている IP アドレスは，初期設定で設定した Wi-Fi ネットワーク上での，Event Learning Sensor のアドレスです．必要になるのでコピーしておきましょう．
2. Wi-Fi 設定を開き，初期設定で設定を行ったアクセスポイントに接続してください．
3. ブラウザを開いて，先ほどコピーしたアドレスに接続してみます．すると，同じ画面が表示されます．この画面から学習の指示などが行えます．

A.5 Event Learning Sensor の LED 表示

Event Learning Sensor には動作を確認するための LED が備え付けられています．LED の点灯状況で，現在の動作を確認することが可能です．学習に移る前に LED の表示を確認しておきます．LED の点灯例は別紙図 A.5 を参照してください．

- オレンジ色 : 認識・学習待機中です．大きな音が発生すると，学習の支持があった場合は学習を，指示がなかった場合は認識を行います．
- 青色，赤色 : 学習済みのイベントを認識しました．青色 LED(上) 側からイベント ID0,1,2,3 のように並んでいます．画像の例だと，青色 LED が点灯している方はイベント ID0 に学習させたイベントを，赤色 LED が点灯している方はイベント ID1 に学習させたイベントを認識したということです．
- オレンジ色＋赤 or 青色 : 点灯しているイベント ID を学習しています．画像の例だと，イベント ID1 に音を学習させているということです．

A.6 学習

画面表示は別紙図 A.6 を参照してください．LED の点灯例は別紙図 A.5 を参照してください．

1. 学習画面の表示で開いたページから，Event Learning Sensor へ学習の指示を送ることができます例のように，イベント ID を選択してイベント名を入力してください．

なお，イベント名は半角英数字のみ利用可能です．イベント名は，アプリの作成でも利用するので，覚えておいてください．(学習画面で確認できます）なお，イベント ID0 を選択した場合は，環境音を再度学習します．

2. 学習の準備が完了しています．イベントを発生させてください．
3. 学習対象の音を発生させた時に，Event Learning Sensor の LED がオレンジ色＋赤色点灯しているか確認してください．学習対象の音を発生させた時に，同時点灯していない場合，誤った音を学習してしまっている可能性があります．その場合は，再度学習の操作を行ってください．なお，再学習する場合はイベント名を入力する必要はありません．
4. 正常に学習されているか，同じ音を発生させて確認することをおすすめします．同じイベントを発生させ，学習させたイベント ID に対応する LED が点灯していることを確認してください．誤認識される場合は，再度学習の操作を行ってください．

A.7　スマートホームアプリケーションの作成

アプリケーションの作成にはインターネット接続環境が必要です．Event Learning Sensor のネットワークに接続したままでは，インターネットに接続できず，アプリケーションの作成がでません．Wi-Fi 設定を変更するか，Wi-Fi を OFF にしてモバイルネットワーク接続をご利用ください．Event Learning Sensor のネットワークに接続せず，学習などの操作を行う方法は,「第 4 章　学習画面の表示」を確認してください．

画面表示は別紙図 A.7 を参照してください．

1. 学習が終わったら，学習結果をもとに動くスマートホームアプリケーションを作ります．IF アプリを開いて，レシピ一覧画面を開きます．(別紙図 1 の (2) に示しているボタンを押します)「＋」のアイコンをタッチします．
2. 「Create a New Recipe」をタップして新しいレシピを作成します．
3. 「if なにが起きたら then どうする」という用に設定します．まずは「何が起きたら」を設定しましょう．
4. 検索を利用するなどして，Maker チャネルを探し出します．Maker チャネルを選択すると画面が切り替わるので,「Receive a web Request」を選択します．
5. イベント名の入力画面に移ります．先ほど学習させたのと同じイベント名を入力してください．入力が終わったら，右上の next を押してください．
6. 次に,「どうする」を設定します.「＋」アイコンを押すと，(2) のような画面になります．この画面から，先ほど Maker チャネルを選択したように，好きなチャネルを選択できます．自由に設定してみましょう．選択したものによっては，アカウント

の連携などの動作が必要になるので，画面の指示に従ってください．例として，スマート電球 Hue の場合の操作を次の章に載せています．

利用できる動作の例）

スマート電球 Hue の操作，スマートフォンへの通知，メールの送信，twitter への投稿，など．

7. 最後の確認画面です．イベントの発生をスマートフォンに通知するかどうかを選択して，Finish をタップすれば，スマートホームアプリケーションの作成は完了です．Event Learning Sensor は，学習が終わった時点からイベントの認識を常に行っています．そのため，スマートホームアプリケーションの作成が完了した時点から，アプリケーションが動くようになっています．学習させたイベントを発生させて動作を確認してみてください．

A.7.1 Hue を利用する場合

「どうする」の部分では，様々なサービスを利用できますが，ここではスマート電球 Hue を利用する場合を例にとって説明します．画面表示は別紙図 A.8 を参照してください．

1. 図 6(7) の「＋」アイコンを押します．一覧から探すか，検索バーを利用して,「Philips hue」を探しだし，選択します．
2. hue で利用できる操作が一欄となって出てきます．この中から，イベントが発生したら Hue にさせたい動作を選択します．一番動作がわかりやすいのは，一番下にある「Toggle lights on/off」です．
3. 初めて Hue を利用する場合は，Hue との連携設定が必要です.「Continue」を押して設定に進んでください．
4. hue のアカウントを取得したことがある場合はログインしてください．hue のアカウントを持っていない場合は，アカウントを取得する必要があるので,「アカウントの作成」を押してください．
5. アカウントを作成します．google のアカウントを持っている場合は，google アカウントで登録でき，非常に簡単です．持っていない場合は必要事項を入力して登録してください．
6. アカウントの登録が終了したら，Hue ブリッジとのペアリングを行います．画面の指示に従って，Hue ブリッジのボタンを押してください．
7. ペアリングが終了したら，IFTTT と Hue の連携を承認します.「はい」を押してください．
8. IFTTT と Hue の連携登録が完了しました.「アプリに戻る」を押してください．
9. IFTTT のアプリに戻りました.「Done」を押してください．

10. 最後に，どのライトに対して，(2) で決めた操作を行うか選択します．今回は「All lights」を選択して，すべての電球が一斉に on/off するようにします．ライトの選択が終わったら「Continue」を押してください．これで設定は完了です．(8) に画面が切り替わります.

A.8　トラブルシューティング

最後に，何か問題が起きた時のトラブルシューティングを掲載します.

- 初期設定や，学習用のページが表示されない場合
  →ページを更新してみてください.
  →入力する IP アドレスが間違っていないか確認してください.
  → Event Learning Sensor のアクセスポイントに接続できているか確認してください.
  →一度，Event Learning Sensor をリセットしてください.
- スマホがインターネットに繋がらなくなった
  → Event Learning Sensor に繋げているとインターネットに繋がりません．Wi-Fi を OFF にしてモバイルネットワークで接続するか，自宅のアクセスポイントに接続してください．初期設定が終わった場合は,「4 章 学習画面の表示」に記載してあるような方法で接続が可能です.
- 誤認識が発生する，またはうまく認識されない
  →再度学習すると良くなる場合があります．なお，再学習する場合はイベント名を再入力する必要はありません.

A.9　取扱説明書-画面例-

試用評価 (1) でユーザから画面例と説明を並べて読みたいので，別紙として用意して欲しいと指摘された．そのため，説明書の別紙としてこの画面例を用意した.

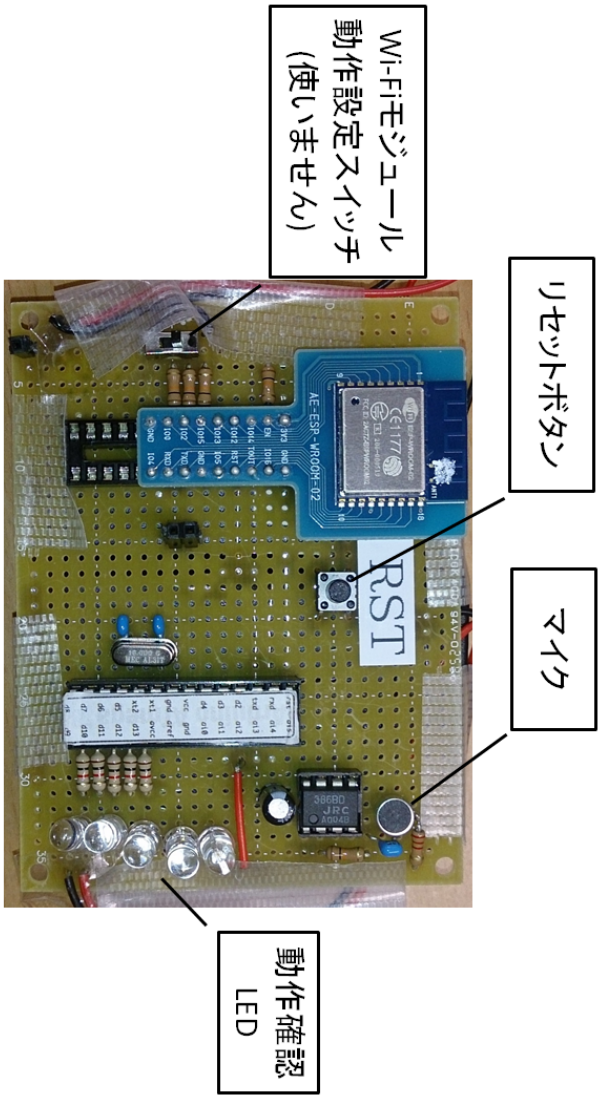

図 A.1: Event Learning Sensor の外観

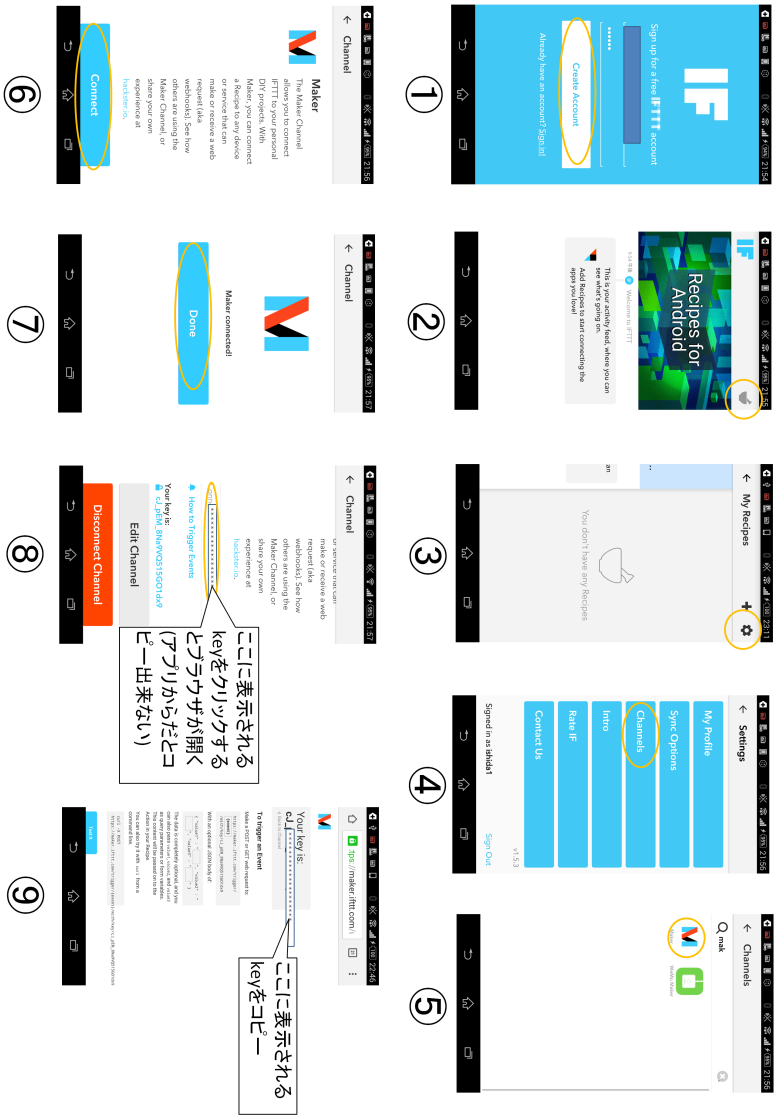

図 A.2: 事前準備の画面例

図 A.3: 初期設定の画面例

図 A.4: 学習画面の表示例

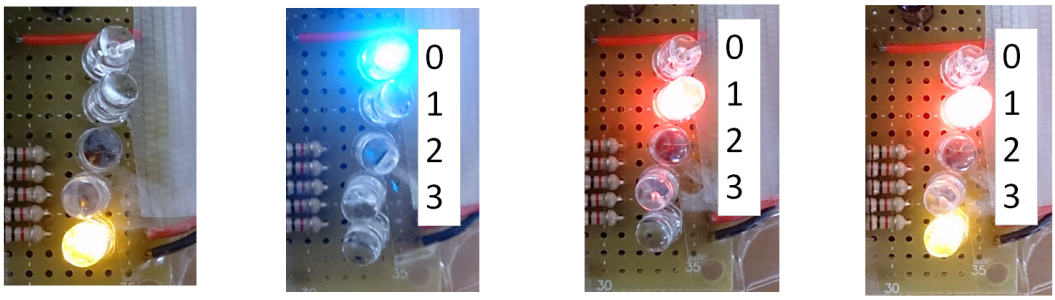

図 A.5: LED の点灯例

図 A.6: 学習の画面例

図 A.7: アプリケーション作成の画面例

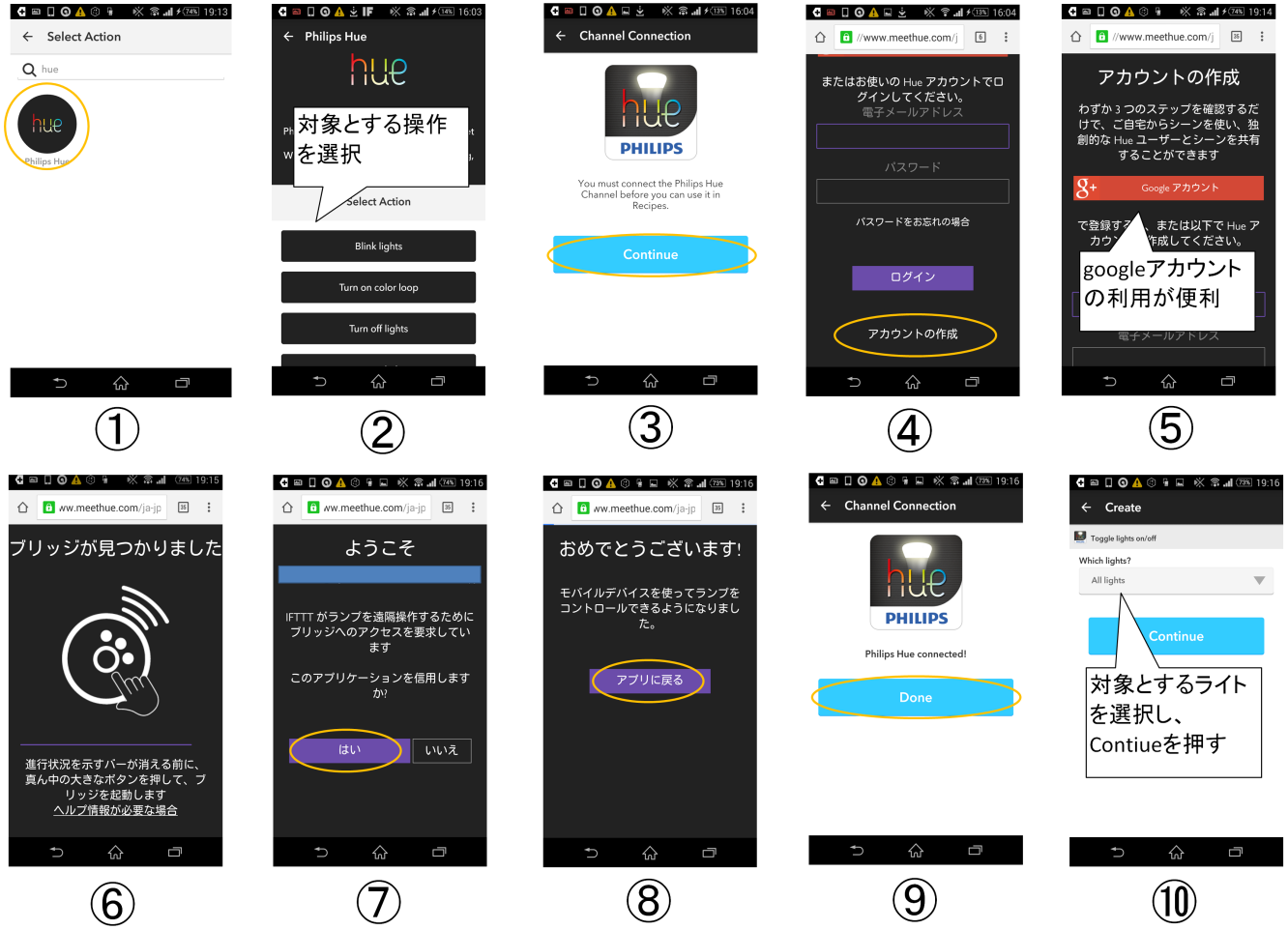

図 A.8: Hue を利用する場合の画面例