筑波大学大学院博士課程

システム情報工学研究科修士論文

光センサ無線デバイスの複数連携によるジェスチャ認識

梶　孝行

修士（工学）

（コンピュータサイエンス専攻）

指導教員　高橋　伸

2016年3月

概要

遠隔で直感的に電子機器の操作を行うため，リモコンの代わりにユーザのジェスチャを利用するアプローチが研究されている．これらは様々な手法が研究されているが，その一つに消費電力の少ないセンサを用いたセンサネットワークを構築する手法がある．しかしこの手法では家の間取りや使用している家電が異なることから，センサの数や位置を調整する必要がある．そこで本研究では，小型で連携可能な光センサ無線デバイスを利用することで，認識したいジェスチャに対して必要なデバイスの個数や位置を容易に試すことのできるシステムを提案する．本研究では光センサ無線デバイスおよびジェスチャの登録可能なシステムの作成を行い，性能評価を行なった．

目 次

図 目 次

表 目 次

第1章　序論

1.1　背景

遠隔で直感的に電子機器の操作を行うため，リモコンの代わりにユーザのジェスチャを利用するアプローチが研究されている．近年ではカメラや kinect 等を利用することで容易にユーザの動きを取得できるため，これを利用してユーザの手の動きを認識して家電機器や大画面操作を行うといったインタラクションが研究されている．しかしこれらの研究の多くはカメラの死角となる部分のユーザの動きを取得することができないため，カメラの正面でのジェスチャに限定されている．これに対して部屋のどの位置にいてもジェスチャにより機器を操作できることができるように，複数のカメラを用いて死角を無くしたジェスチャ認識を行う研究もされている [1][2]．しかしカメラの設置は一般的に防犯や監視として利用される印象が大きい．そのためユーザに監視されている感覚を与えてしまうプライバシーの問題が生じる，また室内にユーザがいる間カメラを起動する必要があり運用コストが大きくなる問題点がある．

一方で，消費電力の低いセンサをネットワーク化することでユーザの行動ログを取得するアプローチが研究されている．これは小型のセンサデバイスを室内の天井や壁に取り付けることで，ユーザがセンサの近くを通ることを認識し，行動ログや電気機器操作に応用している．カメラによるユーザの追跡に対して，ユーザに監視されている感覚を低減した行動ログ取得を行うことができる．

1.2　本研究の目的とアプローチ

本研究では，屋内における電子機器操作のための，プライバシーを重視したジェスチャ認識を行うことを目的とする．また，ユーザごとに認識したいジェスチャは異なると考えられるため，ユーザが簡単にジェスチャを登録，認識できることを目的とする．

本研究では，小型で連携可能な光センサ無線デバイスを利用することで，認識したいジェスチャに対して必要なデバイスの個数や位置を容易に試すことのできるシステムを提案する．これによりデバイスを用いてユーザは認識したいジェスチャが登録できるだけでなく，さらにデバイスを追加することで，認識できるジェスチャを増やす，あるいは他の場所で同様のジェスチャが認識できるように同じようにデバイスを追加で配置するといった使い方ができる．

1.3 本研究の構成

本論文の構成は以下の通りである．本章では屋内における電子機器操作を目的としたジェスチャ認識における問題を挙げ，それを踏まえた本研究の目的とアプローチに関して述べた．第 2 章では，関連する研究に関して述べる．第 3 章では本研究が提案するシステムの説明を行い，第 4 章ではシステムの実装について詳細を述べる．第 5 章では作成したシステムに対する評価と結果について述べ，最後に第 6 章では結論と今後の課題について述べる．

第2章　関連研究

2.1　安価なセンサによるジェスチャ認識

少量の安価なセンサを用いたジェスチャ認識を行う手法として，様々な手法が提案されている．Withana らは三つの赤外線センサを用いてスマートウォッチ上の指のジェスチャを認識する zSense を提案している [3]．zSense では三つの赤外線センサにより画面の直近上の指の動きを取得し，11 種類のジェスチャを認識することができる．また Taylor らはキーボードのキーとキーの間に赤外線センサを 4 × 16 個設置したキーボードを製作し，キーボード上でのピンチ操作やキーボード上空でのホバージェスチャの認識を行っている [4].

さらにセンサをネットワーク化したジェスチャ認識としては Wilson らが天井に設置した赤外線人感センサやドアに接触センサをネットワーク化することでユーザの移動を認識し，照明や空調の操作を行う手法を提案している [5]．Wren らも同様にオフィス内での複数ユーザの直進移動や方向転換，2 人の交差などを検出するシステムを提案している [6]．この研究では赤外線人感センサをセットワーク化しオフィス内に数メートル間隔で格子状に設置して実現している．また本田らも同様に部屋内に赤外線センサを配置して内部の人の行動追跡を行なっている [7].

これらの研究では安価なセンサを少数用いてジェスチャ認識を行う点で同じであるが，本研究ではセンサをネットワーク化することで，ユーザの目的に応じて認識するジェスチャを自由に設定できるシステムを提案する．

2.2　ユーザの影によるジェスチャ認識

精度向上やユーザのプライバシーを守ることを目的として，ユーザの影を用いてジェスチャ認識を行う手法が提案されている．Segen らは手の 3 次元位置を推定しジェスチャ認識を行う Shadow Gesture を提案している [8]．Shadow Gesture では斜め方向から手に対して光を当て，手と手の影をカメラによって取得することで，手に対する影の位置や大きさを取得した映像から得る．この形や大きさを分析することで手の 3 次元位置を推定し，ジェスチャ認識を行う．また Cowan らはプロジェクタの映像を手で塞ぐことによって生じる影によってジェスチャ認識を行う Shadow Puppets を提案している [9]．ShadowPuppets ではプロジェクタとカメラを組み合わせたデバイスを用いて，プロジェクタの映像をさえぎった手の大きさや形から 3 次元的な動きを取得してジェスチャ認識を行う．また Raheja らは天井に取り付けられたカメラの映像から影を抽出し，人物のトラッキングを行っている [10].

本研究も光センサを用いてこれらの研究同様ユーザの影を取得し，ジェスチャ認識を行う．本研究ではこれに加えてセンサネットワークを用いることで，ユーザのプライバシーを重視した上で広範囲でのジェスチャ認識を行う．

2.3 LED の光センサとしての利用

本研究では，光センサとして LED を使用する．LED の光センサとしての利用はいくつか応用がなされており，Dalton は LED マトリクスを使用して入出力を行なう TapTails を提案している [11]．TapTiles は LED をマトリクス状に配置し，光センサとして利用すると同時にドットディスプレイとして利用することができる．これを用いて床にテキストや選択肢を表示して，足による操作を行なうといったインタラクションを行なっている．また秋田らは同様に LED マトリクスを入出力に用いる LEDTile を提案している [12]．LEDTile では LED マトリクスに光を当てることを認識できる組み合わせ可能なデバイスを作成し，オルゴールなど様々なインタラクションに応用している．また Echtler らは液晶スクリーンにおいてマルチタッチを実現するため赤外線 LED マトリクスを利用する手法を提案している [13].

第3章　光センサ無線デバイスの複数連携によるジェスチャ認識システム

3.1　システム全体像

本システムの全体像を図 3.1 に示す.

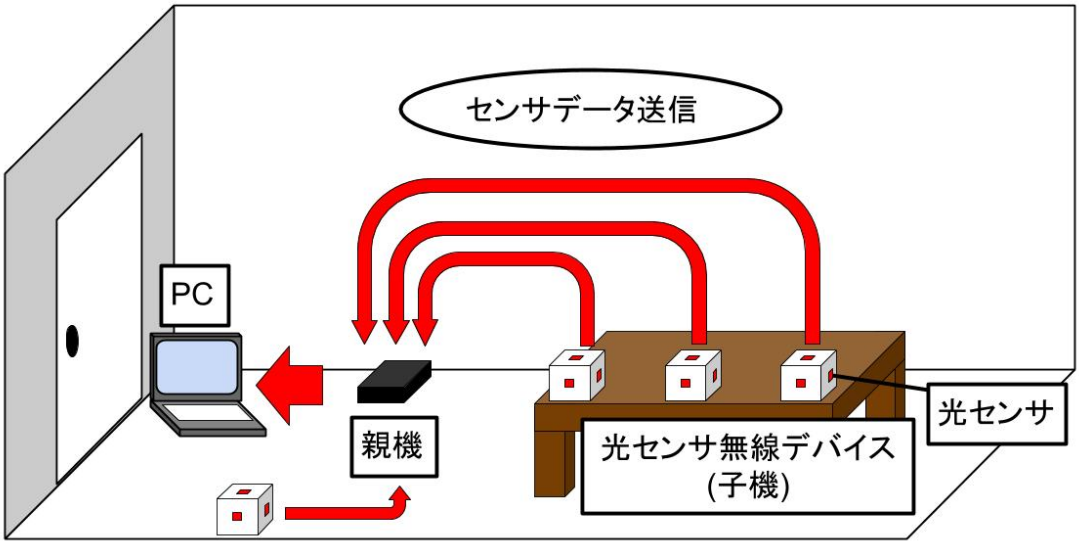

図 3.1: システム全体像

本システムはセンサが取り付けられた無線通信可能な複数の子機およびデータの取得，ジェスチャ認識を行う親機，PC から構成される.

各子機には光センサがついており，ユーザが子機に対して手を近づけたり，子機のそばを通過したりしたときの変化を検知することができる．また子機は無線通信可能となっており，好きな位置に子機を置いた上で親機にセンサ値を送ることができる．親機は PC にセンサ値を送り，PC ではセンサ値を用いることで，ジェスチャの学習，認識を行うことできる.

本システムでは屋内における電子機器の操作のためのジェスチャ認識を目的としている．し

かしユーザごとに操作したい電子機器や期待する操作方法は異なると考えられる．そのため認識したいジェスチャもまた異なるため，様々なケースに対応することが難しい．本システムを利用することで，ユーザは認識したいジェスチャの種類や認識したい場所に応じて子機を部屋に配置し，実際にジェスチャを行うことで学習させることで，期待するジェスチャを認識することができる．また，子機を減らして同様のジェスチャが認識できるか，子機を増やして認識精度が向上するか，子機を増やして扱えるジェスチャの数を増やすといった使い方をすることができる．

屋内における家電の操作のジェスチャとしてテレビの操作を考える．このとき，テレビの前のソファに座ることで電源を入れる，手を左右にスワイプすることでテレビのチャンネルを切り替える，手を円状に回すことで音量の調節を行うなどのジェスチャが考えられる．本システムではこのようなユーザが近くを通る，手を上下左右にスワイプする，円状に回すといった操作を単一の子機で認識するために，子機を立方体の形状とした．立方体の底面以外の 5 面に光センサを取り付けることで，近くを通る際には側面のいずれかのセンサが反応する，子機の上で手をスワイプする際には左側面→上面→右側面のセンサが反応する，手を円状に回すジェスチャに対しては各側面のセンサが時計回りに順番に反応する，のように単一の子機でこれらのジェスチャ認識が可能であると考えられる．

3.2 想定環境

本研究では想定環境として，蛍光灯等の拡散光源が天井に一つから複数存在し壁からの反射光なども存在する部屋におけるジェスチャ認識を対象に考える．想定する環境において光センサは直近から 20cm 程度上空の環境光の変化に対して反応する．

3.3 利用シナリオ

本システム利用の流れは以下のようになる．

- 各子機を認識したいジェスチャに応じて好きな位置におく．
- 認識したいジェスチャを実際に行うことで学習を行う．
- 学習したジェスチャの認識を行う．
- 結果に応じて最初に戻り子機の数や位置を変化させる．
- 登録したジェスチャが認識可能となる

具体的な利用シナリオとして，手を用いた家電操作の例としてテレビの操作を行なうシナリオ，人の通過を用いて照明，空調操作を行なうシナリオを以下に述べる．

3.3.1　手を用いたジェスチャによるテレビの操作

ユーザがテーブルの上でテレビの操作を行なうため，ジェスチャの登録を行なう流れを示す．

1. まずユーザはテレビの電源，チャンネル，音量を操作するために，机の上に一つ子機を設置し，上面のタッチによる電源の ON，左右のスワイプによるチャンネル切り替え，手を円状に回すことによる音量調節のジェスチャを登録し，テレビの操作に割り当て利用する．

2. テレビの操作として番組表をジェスチャで表示するため子機を掴むジェスチャを登録する．

3. 番組表のスクロールでは日付単位や一週間単位のスクロールも行なうため，子機を横に複数並べてスワイプの距離に応じて 1 つでは 1 時間，2 つでは 1 日，3 つでは 1 週間スクロールするようにジェスチャを登録する (図 3.2).

4. ユーザはソファに座った際には常にテレビを見ていたため，ソファに座った際に電源が点くように子機をソファの足元に置き，座った際にテレビの電源を入れるようにジェスチャを登録する．

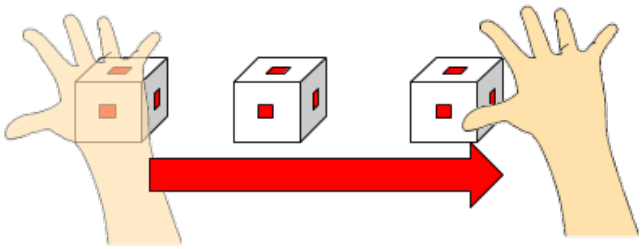

図 3.2: 子機を並べることによるスワイプジェスチャ

このように子機を増やすことで，ユーザが複雑なジェスチャを行いたい際にも認識することができる．また子機を増やすことでユーザに合った操作設定に拡張していくことができる．

3.3.2　人の通過による照明および空調の操作

部屋に入った際に空調や電気が点くようにシステムを利用する流れを示す．

1. ユーザが部屋に入ったことを認識するため，ドアの付近に子機を設置し，実際に入室，退室のジェスチャを登録し，空調，照明に割り当てる (図 3.3).

2. 実際に利用したところトイレなどで一時的に部屋を離れる際に空調が切れてしまい，再入室時に運転音が気になることが判明する．

3. 玄関および寝室のドア付近に子機を設置し，玄関を経由して部屋に入った場合，目が覚めて寝室から部屋に入った際にのみ空調の電源を入れ，反対の行動をすることで空調を切るようにジェスチャを再登録する．

4. トイレなど一時的に部屋を離れる際では空調はそのままで照明のみ消え，外出，就寝時では照明および空調が切れるようにジェスチャを認識することができた．

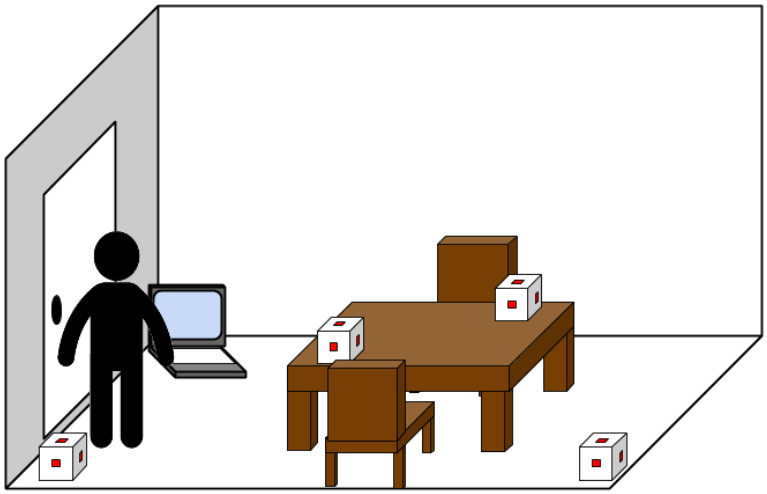

図 3.3: 人の通過によるジェスチャ

このようにジェスチャ登録後にユーザの期待と異なる動作を行なってしまう場合でも，子機を追加することで，ユーザが期待する状況でのみジェスチャが認識されるようにカスタマイズすることができる．

第4章　実装

4.1　開発環境

本研究では光センサ無線デバイスおよびジェスチャ認識システムの作成を行った．デバイスは無線通信用に XBee を使用し，その設定ソフトウェアとして XCTU を用いた．またデータの加工を行うために，Arduino を使用し，開発言語にも同様に Arduino を使用した．ジェスチャ認識システムには開発言語に C#を，開発環境に Visual Studio2012 を使用した．

4.2　デバイスの実装

本研究では光センサ無線デバイス (子機) およびデータを集める親機を作成した．それぞれについて以下に述べる．また本デバイスでは無線通信を行うために，ZigBee と呼ばれる近距離無線通信規格を用いる．製品としては XBeeZB モジュール (XB24-Z7PIT-004) を使用した．ZigBee は転送距離が短く転送速度も低速であるが，扱いやすく安価で消費電力が小さい特徴を持つ．データ転送速度は 2.4GHz で 250Kbps であり，簡単な設定でセンサネットワークを構築することができる．

XBee はネットワーク内に一つ設定できる Coordinator と中継機能をもち通信可能距離を伸ばすことできる Router，中継機能を持たないがスリープ設定ができ省電力で長期間稼動させることのできる End Device の設定が可能である．本システムでは屋内にセンサを配置してセンサ値をとる通信を想定している．そのため XBee の通信距離である 40m を超えることは多くないと予想されるため，親機に用いる XBee は Coordinator として，子機には End device の設定を用いる．

XBee はそれぞれ別々の 64 ビットアドレスを持っており，通信時にはこれを用いて送信先や送信元を管理する．本研究では子機として用いる XBee にそれぞれ ID を 1 から割り振り，この ID を元にデータの管理を行う．そのため複数利用時にはこの順に使用するものとする．

また XBee の通信に 1 対 1 で通信を行なう AT モードと 1 対多で通信を行なうための API モードの 2 種類が存在する．本システムでは親機に対して多数の子機のデータを送るため API モードを使用する．API モードの通信では，データを API フレームという独自のプロトコルでパケット化して通信を行う．これによって文字列の送信だけでなく XBee の操作 (ピンの ON，OFF 指定やスリープの ON，OFF 指定など) が可能となる．

API フレームは大まかには以下の構成になっている．

- 開始コード (0x7E)

- フレームデータの長さ (2byte)
- フレームデータ
- チェックサム (0xFF - (データの総和 & 0xFF))

まず受信側では開始コードが一致しないデータは無視し，開始コードが来た場合データの長さを取得し，その分受信を行なう．その後フレームデータの長さ，フレームデータ，チェックサムを受け取りチェックサムを用いてフレームデータに通信時の欠損がないかを調べる．

実際に送られる API フレームの例を図 4.1 に示す．

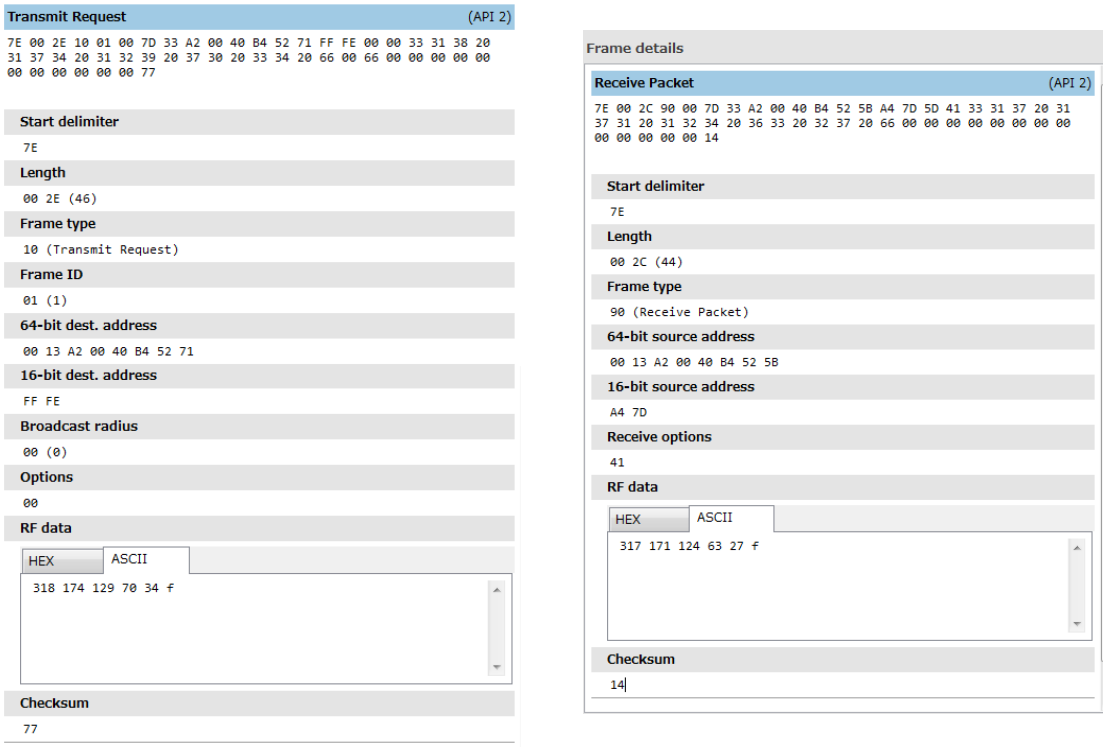

図 4.1: API フレーム 左: 送信側 右: 受信側

図上部の Transmit Request は実際に送られるデータであり，その下には各構成の内容が個別に表記されている．送信側では先の構成に加えて，フレームデータとして文字列送信であることを示すフレームタイプや送信先アドレス，送信する文字列などが含まれている．受信側では文字列受信であることを示すフレームタイプや送信元アドレス，送られてきたデータなどが含まれる．Arduino から XBee との通信には公開されている XBee 用ライブラリを使用した [14].

4.2.1 親機の実装

作成した親機を図 4.2 に示す.

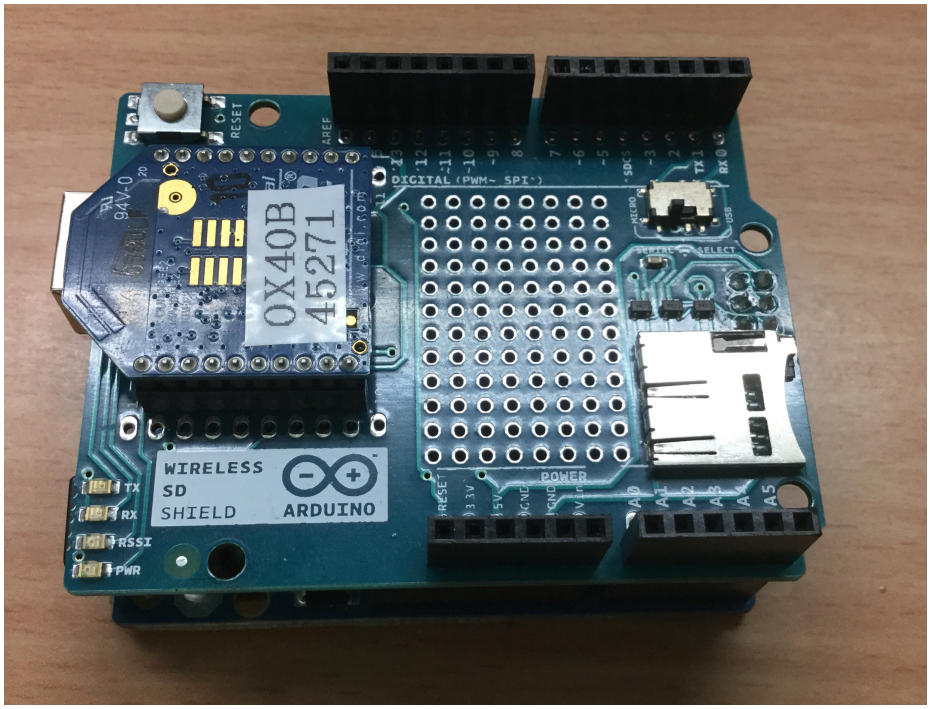

図 4.2: 作成した親機

親機は子機からのデータを受け取るための Arduino uno と XBee 用シールドである Arduino ワイヤレスプロトシールドから構成される．PC と Arduino は USB 接続し，シリアル通信によって子機から得たセンサ値の送信を行う．親機では使用する子機のアドレス別の ID を管理し，これを用いて LED のデータを別々に管理する．親機は以下のプログラムの流れで動作する．

1. 子機からのデータを受け付ける
2. データが送られた際に子機のアドレスから ID を取得し，該当するセンサ値を更新する．
3. 20ms ごとに最新の各センサ値を空白区切りで文字列にまとめ，PC にシリアル通信で送信する．
4. 1 に戻る．

また親機の Arduino は XBee を介して子機との通信を，USB を介して PC とのシリアル通信を同時に行う．これに対してシリアル通信は単一のピンで行うため，プログラム上で文字列を送信した際に，XBee および PC 両方に同じ文字列が送られてしまう．そのため送られた

文字列に特徴を持たせて区別する必要がある．XBee との通信は独自の API パケットで行ない「0x7E」で始まるパケットのみを受け付けるため，PC との通信を区別することができている．PC との通信の際には API と違いを持たせるため，最初に [7E] という文字列を付与する．これによって子機との通信と PC との通信を区別し，同時に動作できるようにする．

4.2.2 子機の実装

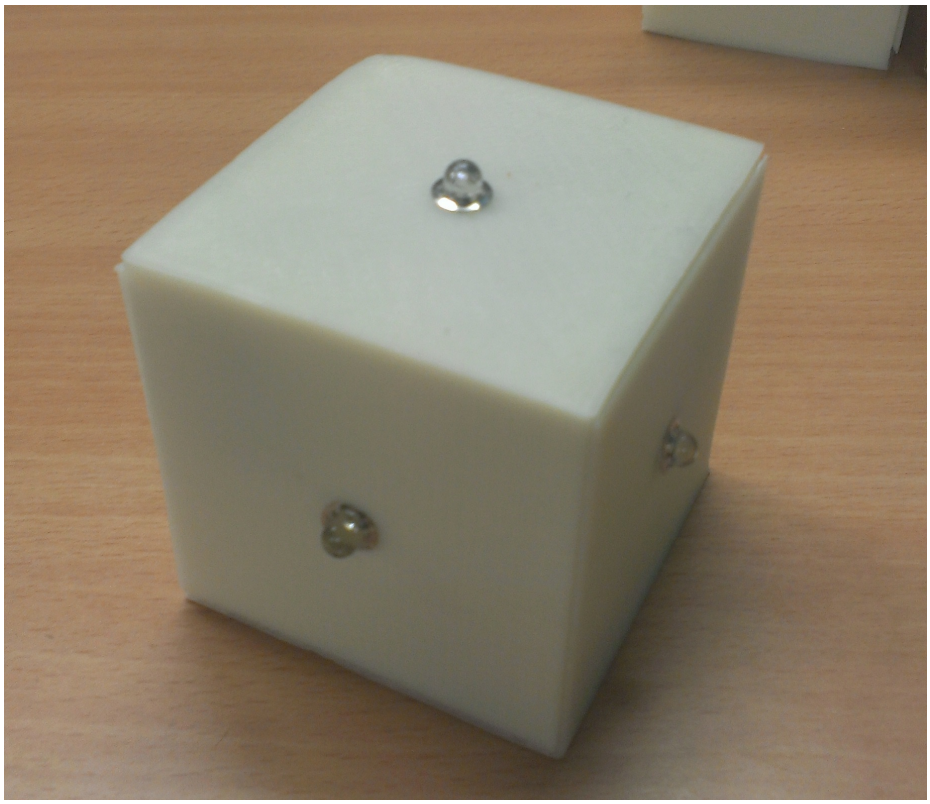

図 4.3: 作成した子機

子機は，無線通信を行うための XBee，センサの値を読み取るための Arduino uno，そして光センサから構成される．Arduino uno はマイコン部分である ATMEGA328P-PU に水晶発振機およびセラミックコンデンサを接続し，適切なブートローダを書き込むことで 3.0V 前後の電圧で動作させることができる．ブートローダの書き込みには Arduino uno を用いて行なう ArduinoISP スケッチを使用した．書き込み先は「Arduino Uno，Arduino Pro or Pro Mini(3.3V，8MHz)w/Atmega328」を，書込装置には「Arduino as ISP」を使用した．これにより省スペース化とコスト削減が行えるため，上記の素子をユニバーサル基盤上に実装した．

また本研究には光センサとして LED を使用する．LED は電流を流すことで発光する電子素子であるが，反対に光を当てることによって微量の電流が生じる．そして Arduino のアナロ

グ入力に繋ぐことにより，LED に対する光の強さによって 0〜1.2V 程度の電圧値を得ることができ，ある程度光センサとして利用することができる．また一般的に可視光に対して利用される光センサには CdS 等が存在するが，LED を利用することによって発光という出力と光センサとしての入力が単一の LED で行えるため，ユーザへのフィードバックなどへの応用が期待出来る．

作成した子機の全体像は図 4.3 のようになる．

子機は一辺が 60mm の立方体の形状であり，底面以外の 5 面の中心に LED が露出するように取り付けられている．

子機のケースは 3D プリンタによって作成を行った．3D プリンタにはダヴィンチ 2.0ADuo(図 4.4) を，3D モデルの作成には AUTODESK の 123D Design を使用した．

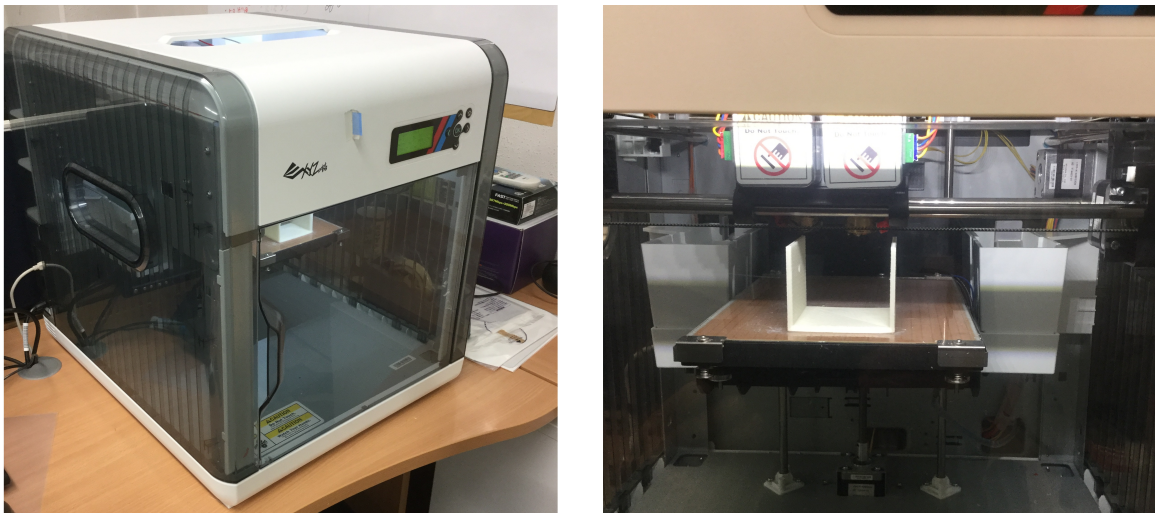

図 4.4: 使用した 3D プリンタおよびケース作成の様子

積層型 3D プリンタは仕様上空中で水平方向の面を作成したり複雑な形を印刷することが難しい．そのため図 4.5 のように 3 面ずつの部品を組み合わせる形でケースを作成した．作成したモデルは端が 45 度の傾斜となるように削られている．ここを図 4.6 のように組み合わせることによって立方体の形状となる．そしてケースの内部の空間に基盤と電源として単 4 電池二本および LED を収納する．

子機の基盤回路は図 4.7 のようになる．

図 4.7 の下部分は ATMEGA328P-PU である．駆動のため VCC および AVCC のピンに電源である 3.0V を接続し，XTAL1，2 ピンに 16MHz の水晶発振機および 22pF のセラミックコンデンサを接続する．そしてシリアル通信のため RX，TX ピンを XBee のシリアル通信用ピンと接続する．アナログ入力として利用できる A0〜A5 ピンには LED と発光させるため 470 Ω の抵抗を接続する．また XBee とのシリアル通信のために GND の共有を行う．

ユニバーサル基板上には図 4.8 の配置で実装を行う．実際に実装した基盤は図 4.9 である．Arduino や XBee は書き換えが可能なようにソケットを用いて接続している．またケースが立

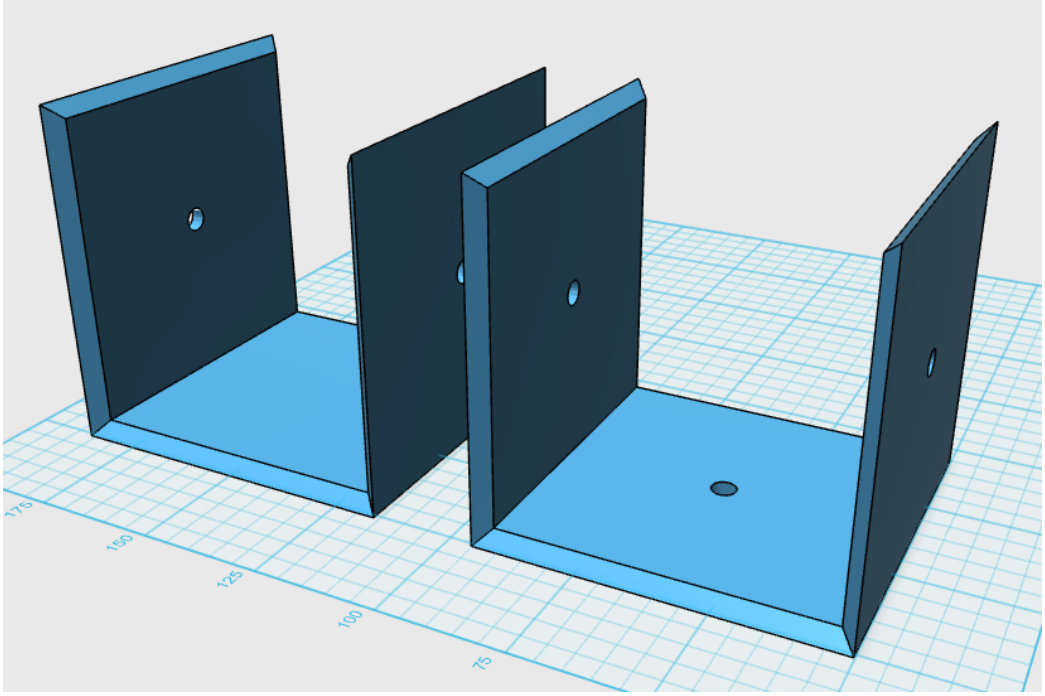

図 4.5: ケース用 3D モデル

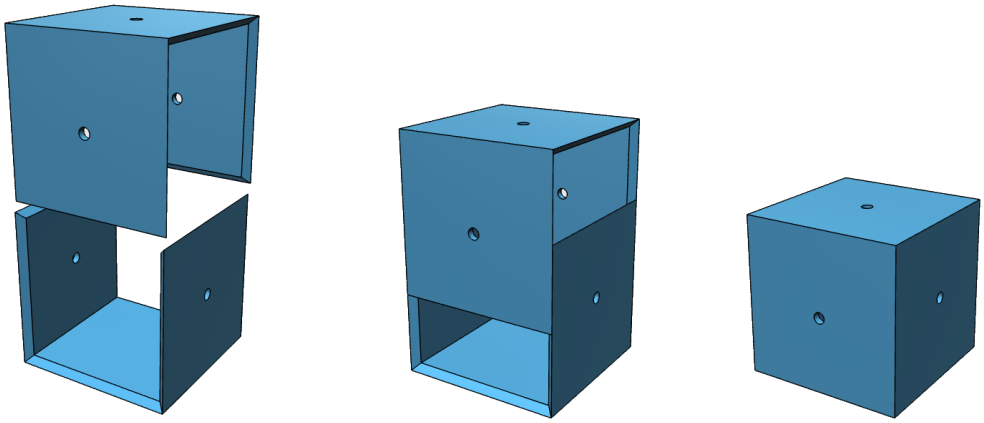

図 4.6: ケースの組み合わせ

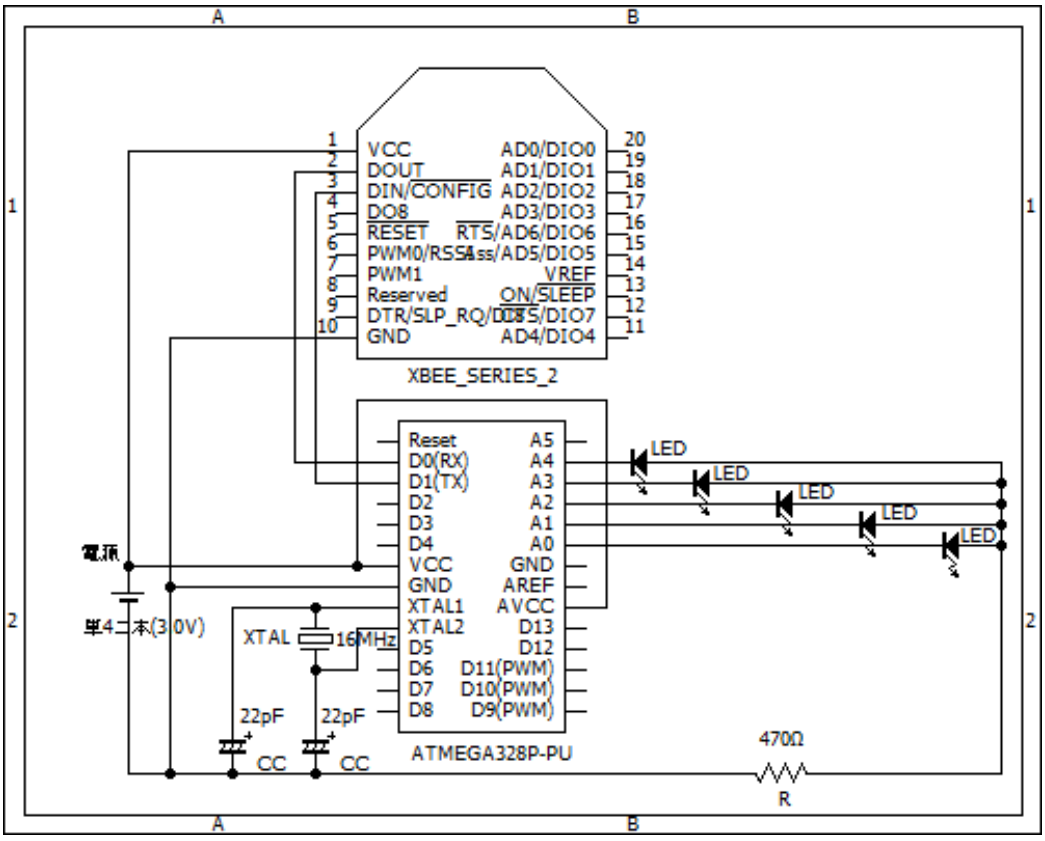

図 4.7: 光センサデバイス回路図

方体の形状でスペースを有効に利用するため，XBee は Arduino の上に設置する．これによって基盤の大きさを抑えるとともに配線の単純化を行った．LED は空中配線でケースに接続し，ブラケットを用いてケースに固定する．

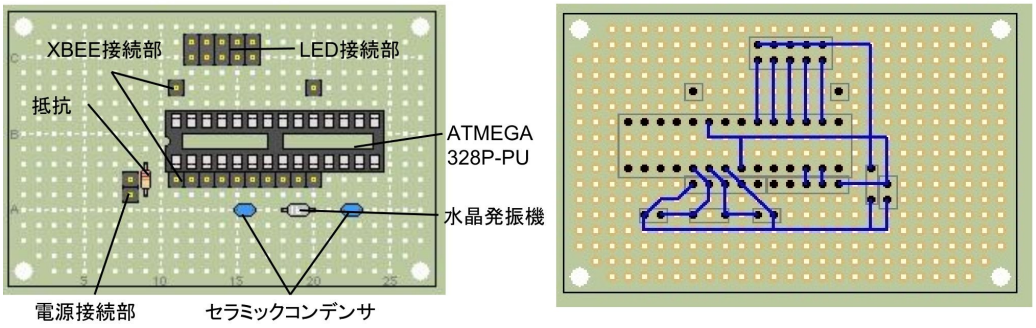

図 4.8: 基盤回路図

Arduino の動作としては以下の動作を繰り返す．

1. 各 LED のセンサ値を取得して保存する．
2. 各 LED のピンを出力に切り替え，点灯あるいは消灯させる．
3. 平均化するため，1，2 の操作を一定回数繰り返す
4. 平均したセンサ値を空白区切りで一つの文字列にまとめる
5. XBee へシリアル通信で 4 の文字列を送信する．
6. 1 に戻る

Arduino のアナログ入力ピンは出力としても利用できるが，入力と出力を同時には行えない．そのため 1，2 のように入力と出力を高速に切り替えることによって，残像効果により出力と入力が見かけ上同時に行えるようにしている．また 20ms の周期で XBee に文字列を送ることができる．

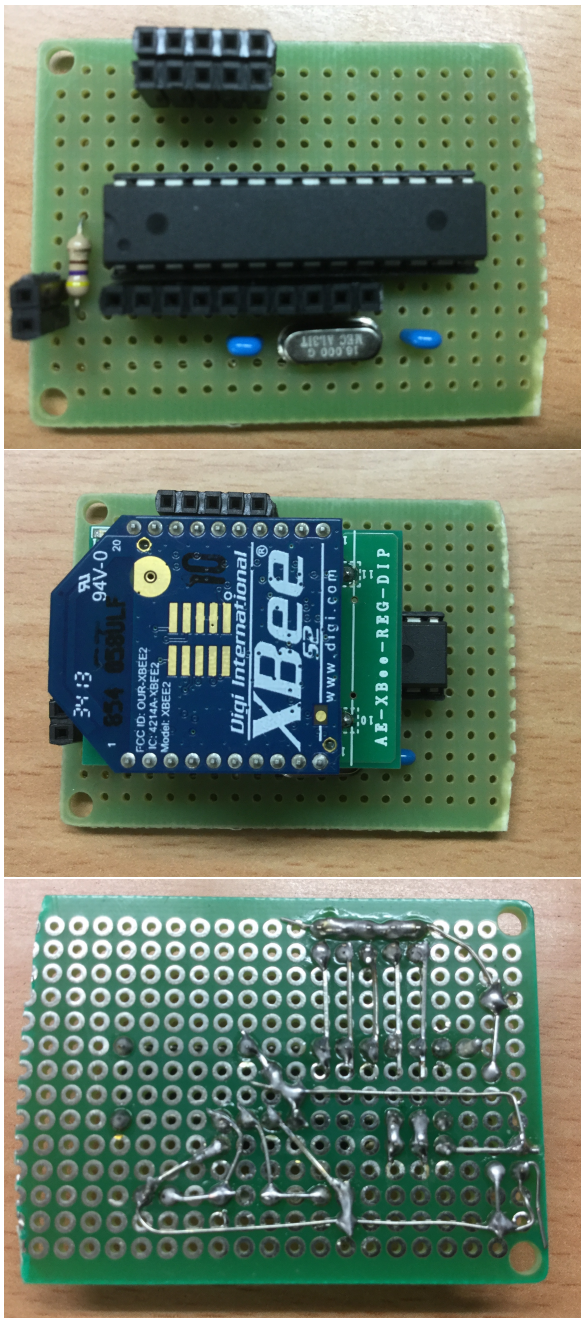

図 4.9: 作成した子機回路 上: 表面 (Arduino 接続状態)　中: 表面 (XBee 接続状態) 下: 裏面 (配線)

4.3　ジェスチャ認識システムの実装

本研究ではジェスチャの登録と認識が可能なシステムの作成を行った．デバイスからのデータの扱い，およびジェスチャの登録，認識についてそれぞれ以下に述べる．

4.3.1　親機からのデータ取得

本システムでは各子機のセンサ値を画像として表示を行う．Arduino では 0〜1.1V までの電圧を 0〜1024 の値で計測する．実際に照明条件を想定環境において，照明の真下や近くにセンサを配置したり，手で影を作るといった操作を行なった際のセンサ値を測ったところ LED から得られるセンサ値は 100〜600 程度の値であった．これを画素値である 0〜255 に割り当てるため，式 (4.1) によって計算を行う．

$$ret = (x - 100)/500 \times 255 \tag{4.1}$$

式 (4.1) 中の x は Arduino によるセンサ値，ret は変換後の値である．

親機からは空白区切りで各 LED のセンサ値が ID 順に送られてくる．システムではこれを分解してそれぞれの値を整数型に変換し，List 構造を用いて値を保持する．画像を表示する際には，List から Bitmap に変換することでグレースケール画像を作成する．しかし，C#の pictureBox では数ピクセルの画像の表示では表示時の拡大操作においてアンチエイリアシング処理がなされ，図 4.10 のようにぼけた画像が表示されてしまう．

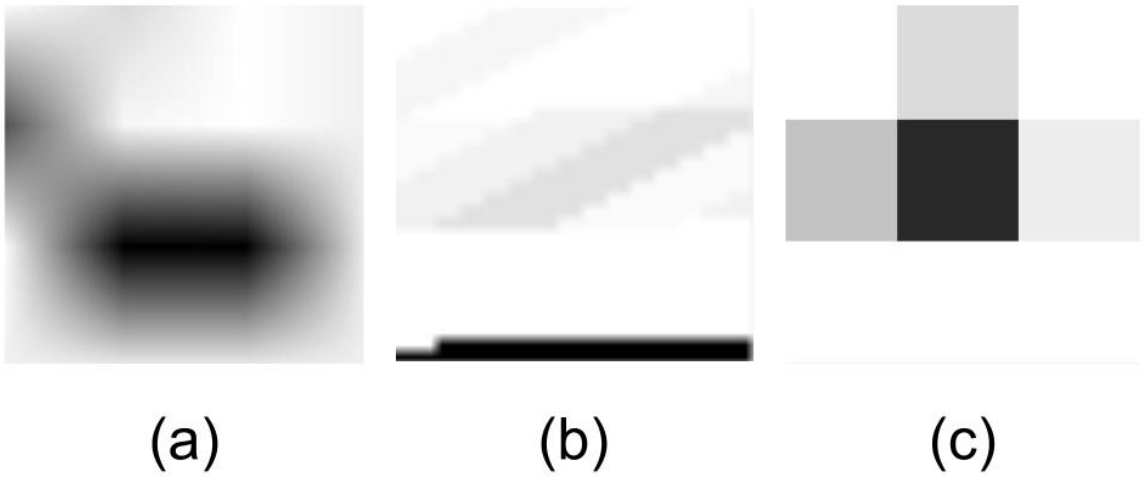

図 4.10: 影画像の表示 (a) 3 × 3 (b) 30 × 30 (c) 300 × 300

図 4.10 は 9 つのデータを 3 × 3 の画像に変換して表示を試みている．図 4.10(a) は 3 ピクセル × 3 ピクセル，図 4.10(b) は 30 ピクセル × 30 ピクセル に拡大を行なって表示しており，どちらも表示が崩れていることがわかる．図 4.10(c) は 300 ピクセル × 300 ピクセル に拡大した

のちに表示しており，境界線がはっきりと表示できていることがわかる．これから数百ピクセルの画像に拡大処理を行い表示する．

$H \times W$ の画像を作成する際，各ピクセルは式 (4.2）を用いることで該当する要素にアクセスできる．

$$id = (h \times col/H) \times row + (w \times row/W) \tag{4.2}$$

h，w はピクセルの座標で col，row はそれぞれ元のデータの縦幅，横幅，id は該当する List のインデックスである．

各 LED は個体差や配置している環境によって，何もしていない際の値がそれぞれ異なる．この状態では識別が困難であるため，LED に対して何もしていない際の値を保存しておき，これを基準として影が生じた際の変化を引いた差分画像 (以下，影画像) を式 (4.3) 作成して，これを認識に用いる．

$$SI_{ij} = 255 - (Base_{ij} - Newimg_{ij}) \tag{4.3}$$

SI は作成する影画像，Base は基準画像，Newimg は最新の画像であり，この計算を全ピクセルに対して行う．

図 4.11 に Newimg，Base 画像，影画像の例を示す．

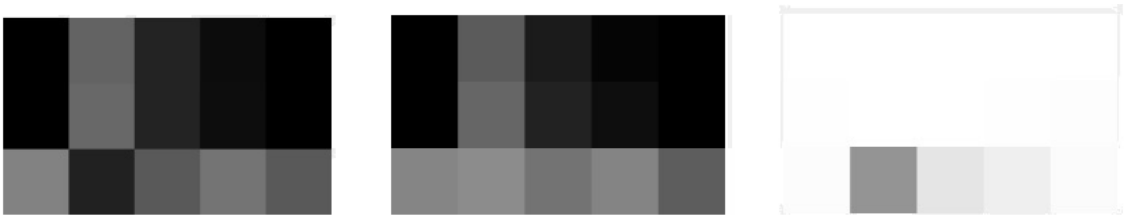

図 4.11: 影画像例 左: Newimg 中: Base 画像 右: 影画像

図 4.11 は何もしていない際の画像を Base として登録した後に，ID3 の子機の正面に対して手を近づけた際のそれぞれの画像を示している．新しく取得された Newimg の子機 1，2 のようにセンサに手を近づけていなくても個々の LED の値には差があることがわかる．また最終的に得られる影画像には現在反応する物体があるセンサのみが黒く表示されていることがわかる．

また影画像から手を振るといった動きの認識を行うためには，一定期間の影画像に対してそれぞれ認識を行うことで手の動きを取得する必要があり，認識が困難である．そのため本システムでは MHI(Motion History Image) の考え方を用いる [15].

MHI は一定時間の複数の影画像に対してそれぞれ重み付けを行って足し合わせた画像であり，ある時間 t における MHI_t は，式 (4.4) によって計算される．

$$MHI_t = \Sigma_{i=0}^{K-1} w_{t-i} I_{t-i} \tag{4.4}$$

式 (4.4) において K は MHI の作成に使用する影画像の枚数であり，I_t は時刻 t における影画像を表している．また w_t は重み付けを表しており，式 (4.5) によって計算される．

$$w_i = \frac{2(K-i-1)}{K(K-1)} \tag{4.5}$$

作成される MHI のイメージを図 4.12 に示す．

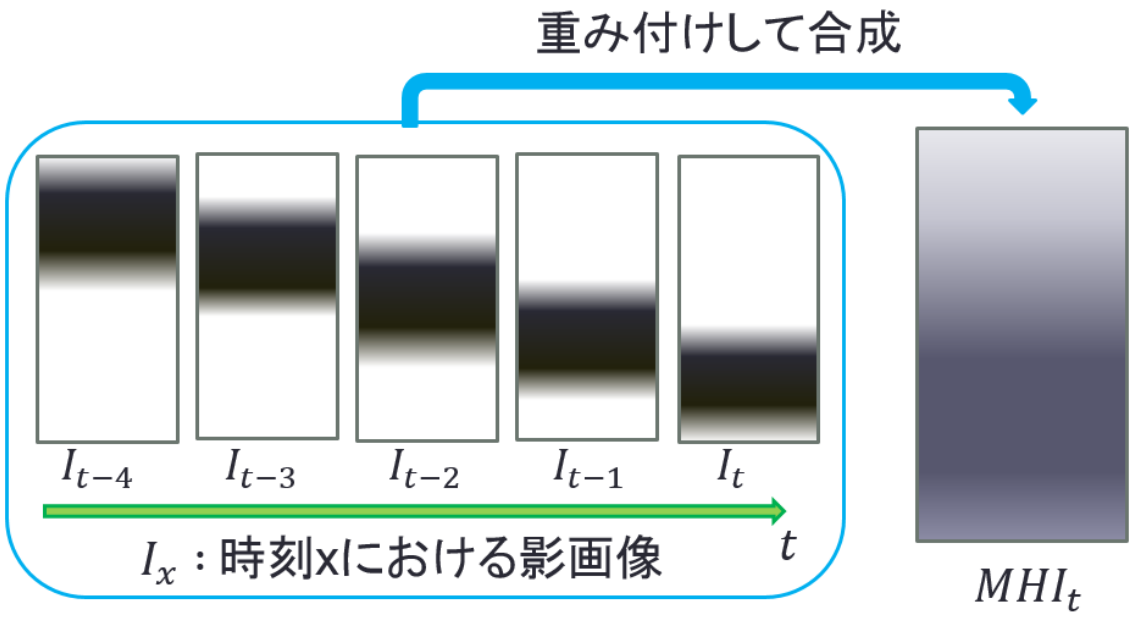

図 4.12: 影画像から MHI の作成

青枠で囲まれた画像は MHI 作成の基となる画像であり，緑の矢印に沿って時系列順に並んでいる．この影画像では横一列に並んだ子機に対して手をスワイプしていくことで得られる影画像をイメージしている．作成された MHI には現時点でのセンサ値の情報だけでなく，過去の情報がある程度含まれる画像となる．そのため青枠の 5 枚の影画像から状態遷移の判定を行なうといった複雑な処理を行なうことなく，1 枚の MHI に対して識別を行うことで動的なジェスチャの認識を行うことができる．

プログラムでは複数の影画像を List で持つことで管理する．新しい影画像が生成された際には List の先頭に追加すると同時に末端の影画像を取り除くことで，直近の指定された枚数の影画像を管理することができる．そして式 (4.4) を用いることでその時点での MHI を作成し，画面への表示，ジェスチャ認識へと利用する．

4.3.2　ジェスチャの登録および認識

ジェスチャの学習，認識システムの各 UI を利用の流れに沿って説明する．本システムの起動画面を図 4.13 に示す．また各機能を表 4.1 に示す．

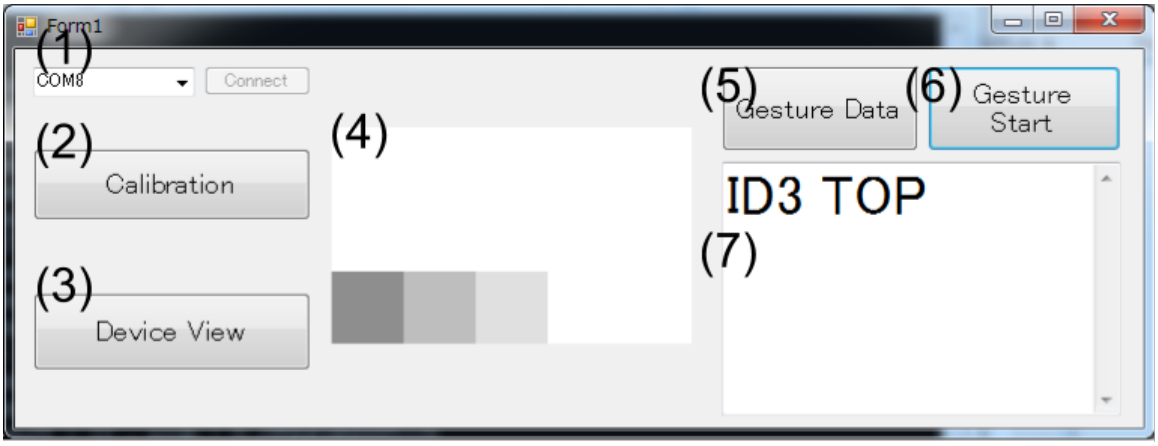

図 4.13: システム起動画面

(1) に接続されているシリアルポートの一覧が表示されるので，親機を指定して Connect ボタンを押すことで接続を行う．(2) を押すことでキャリブレーションを行いセンサ値が (4) に画像として表示される．

画像は 5× 子機のピクセル数となっており，各ピクセルがそれぞれのセンサ値を示している．また各行は各子機で分かれており，接続された 5 つの LED のセンサ値が横に並んで表示される．

また (3) を押すことで図 4.14 のような各子機の個別ビューが生成される．各子機のセンサ値を個別に見ることで，配置が悪いなどの理由で異なるジェスチャを行なった際に似たような結果が返ってくる場合や，ジェスチャに対して反応していない不要な子機などを判断することが期待出来る．

図 4.14 中の SELECT DEVICE で各子機の ID を選択することで，上の表示がそれぞれの子機になる．表示は十字架の画像となっており，中心部分が子機の上部分の LED，そして 4 方に側面の LED のセンサ値を表示している．これは子機の底面を除いて展開した配置と一致しているため，画面を見ながらセンサがどの範囲で反応するのかを確かめることができる．また (3) を押すことで複数表示が可能なため，注視したい子機のみを選択して表示することがで

表 4.1: システム起動画面機能

番号	機能
(1)	親機接続
(2)	キャリブレーション
(3)	子機個別ビューの表示
(4)	センサ値の画像表示
(5)	ジェスチャ登録画面の表示
(6)	ジェスチャ認識の開始
(7)	認識結果の表示

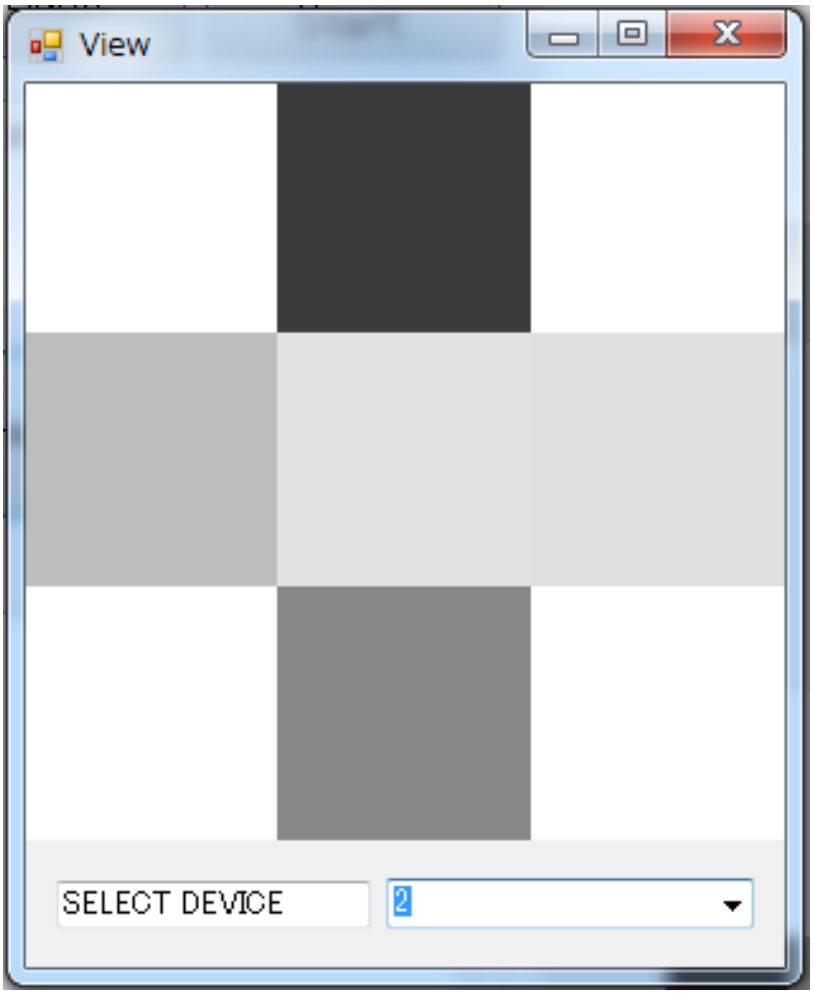

図 4.14: 子機個別ビューの表示

きる (図 4.15).

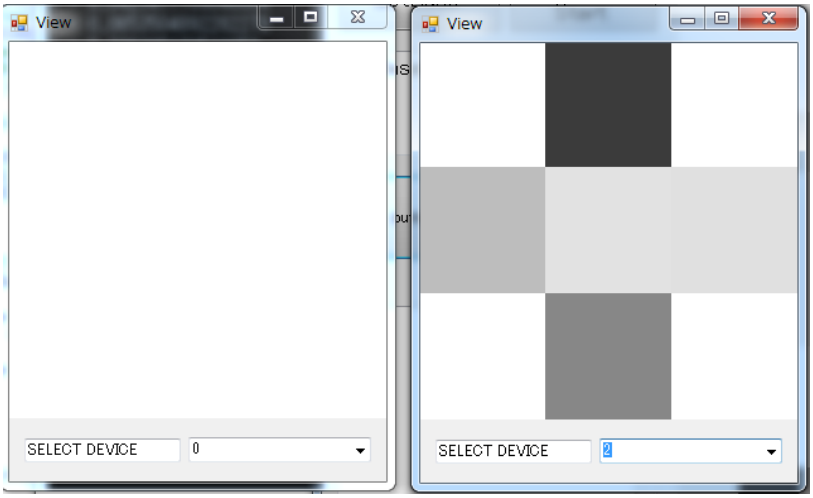

図 4.15: 子機個別ビューの複数表示

また (4) を押すことで，ジェスチャの登録および登録したジェスチャの影画像の確認と誤入力したデータの削除を行なうことができる画面が表示される (図 4.16)．また各機能を表 4.2 に示す.

表 4.2: 登録ジェスチャ確認画面機能

番号	機能
(1)	ファイル指定と読み込み
(2)	影画像の選択
(3)	ID およびラベルの表示
(4)	新しいジェスチャの登録および削除
(5)	影画像の表示

図 4.16(1) の「File Name」にジェスチャ登録用の csv ファイルを入力し，connect ボタンを押すことで指定されたファイルを読み込む．このとき事前にファイルに登録されているジェスチャがある場合は追加でジェスチャを登録することができる．登録済みの影画像がある場合には図 4.16(5) に影画像が表示され，図 4.16(3) に現在表示している影画像の ID およびラベルが表示される．また図 4.16(2) のボタンを押すことで表示する画像を一つ先に，あるいは手前に変更できる.

図 4.16(4) ではジェスチャの登録および削除が可能である.「INPUT LABEL」と表示されているテキストに追加したいジェスチャの名称を入力し，実際にジェスチャを行なった後に SAVE ボタンを押すことで，追加することができる．追加した際には図 4.16(5) に追加した影画像が表示される (図 4.17)．ジェスチャの認識のため，同様のジェスチャを 5 回程度行い保存することが望ましい.

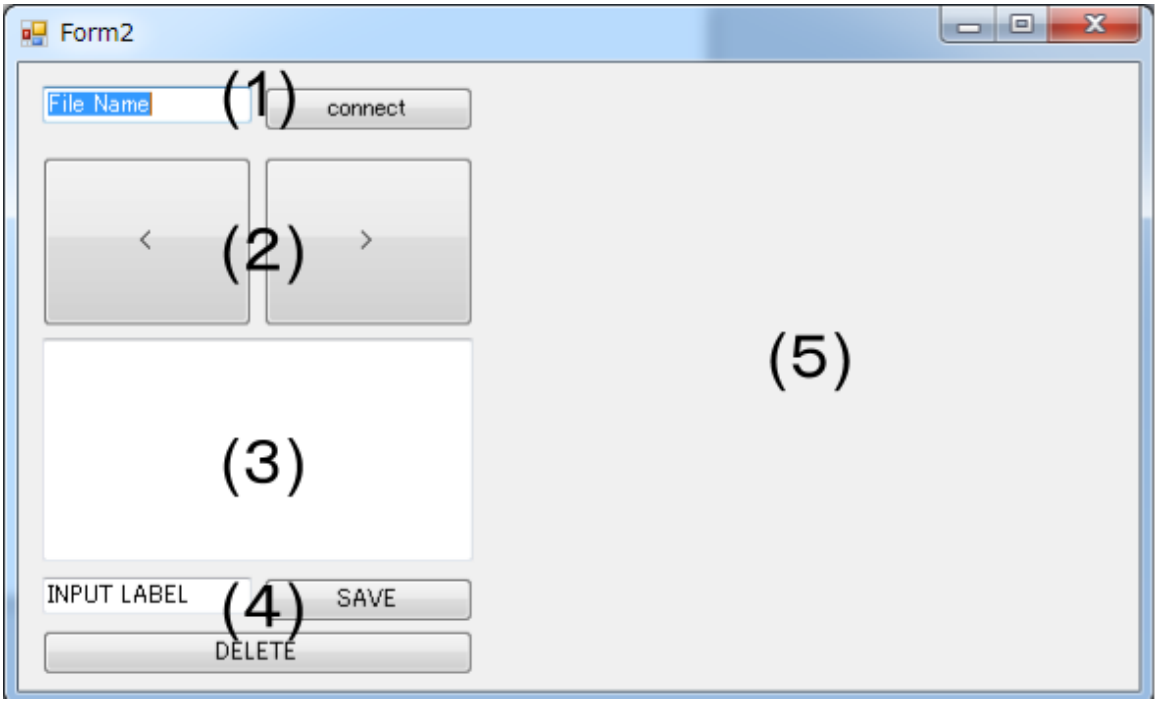

図 4.16: 登録ジェスチャ確認画面

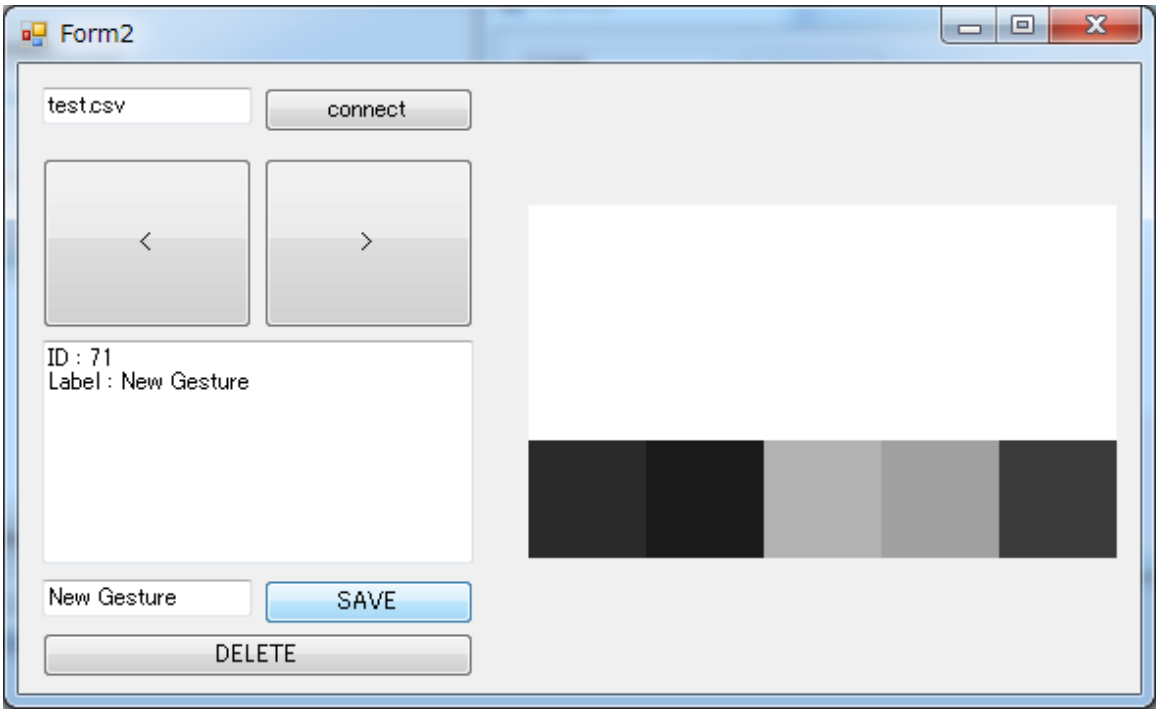

図 4.17: ジェスチャの登録

またジェスチャの登録を誤って行なってしまった，ジェスチャを取り消したい際にはDELETEボタンを押すことで，現在表示している影画像を削除することができる．

システムではユーザが実際にジェスチャを登録する際，ジェスチャ名とそのときのMHIを取得し，CSV形式で保存を行なう．実際に保存されたCSVの例を図4.18に示す．

```
1 Default,253,258,258,259,257,258,258,254,253,256,256,254,249,260,254↓
2 Default,254,254,255,256,256,258,257,255,255,257,256,255,251,257,254↓
3 Default,251,254,257,256,255,254,256,257,255,254,256,253,251,259,254↓
4 TouchFrontC,251,255,255,256,256,257,256,253,255,256,257,103,219,243,256↓
5 TouchFrontC,252,255,254,255,256,258,256,253,255,256,257,98,216,241,254↓
6 TouchFrontC,248,253,254,253,252,257,254,257,256,255,258,101,227,247,260↓
7 TouchFrontC,253,253,253,254,254,261,256,255,256,257,257,95,222,247,257↓
8 TouchFrontC,251,253,255,254,253,256,255,257,255,255,256,97,217,241,254↓
9 TouchFrontC,253,256,260,260,257,252,255,257,253,254,258,68,210,244,258↓
0 [EOF]
```

図4.18: ジェスチャ影画像のCSV保存

ファイルは各行に一つのジェスチャが登録されており，先頭にジェスチャの名称，続いてMHIのデータをそれぞれカンマ区切りで保存している．ジェスチャの追加や削除は，図4.17でファイルを読み込んだ際に，各データをListで保持する．表示する影画像は位置を管理するIDを持つことで行ない，新しいジェスチャの追加，削除はLis tに対して行なう．そして画面を閉じた際に現在保持しているListを読み込んだファイルに上書きする．

ジェスチャ登録後は，図4.13の(6)を押すことで登録したジェスチャを用いて実際に認識することができる．

ジェスチャの認識にはSVMを用いた．実装にはlibsvm[16]をC#用のライブラリとして利用できるLIBSVM.NET[17]を利用した．SVM-typeにはC_SVCをKernel-typeにはRBFを用いた．システムではジェスチャ認識開始時に保存したCSV内のデータを読み込み，これを教師データとして学習を行う．SVMでは学習したデータに対して新しいデータがどのデータクラスに近いかの結果が得られる．そのため，登録したジェスチャ以外の動作を行なった際にもいずれかのジェスチャと認識されてしまう．そのため，ジェスチャを行なっていないデータとして別にランダムに作成したMHIをジェスチャ名NONEとして同時に読み込む．学習後は，新しく生成されたMHIに対して認識を行い，認識結果のジェスチャ名を表示する．

登録したジェスチャと同じ動作を行なうことで，認識ができた場合は図4.13の(7)に登録したジェスチャの名称が表示される(図4.19)．図4.19は図4.17で登録したジェスチャを実際に行い，認識された際の画面である．

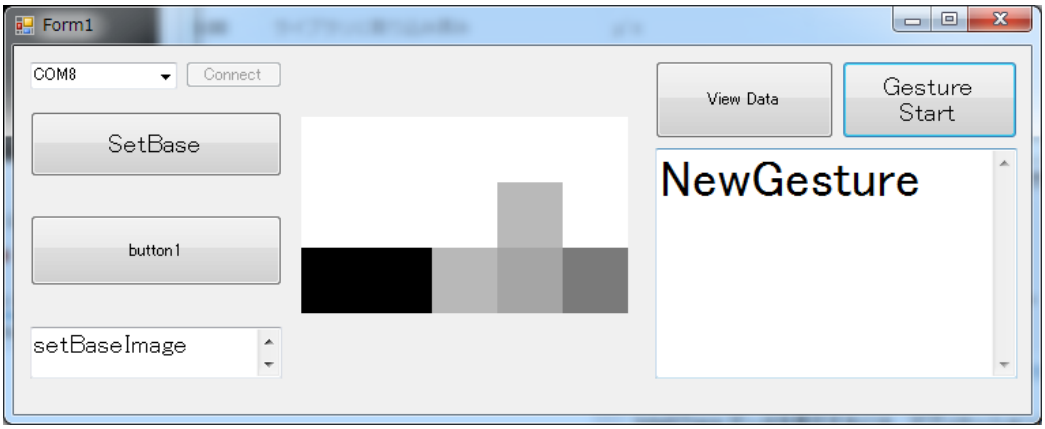

図 4.19: 登録したジェスチャの認識

第5章　評価

5.1　実験1

作成したシステムの性能評価を行なった．実験内容と結果及び考察について以下に述べる．

5.1.1　実験内容

本システムの性能評価として，子機二つを用いて図 5.1 のジェスチャの学習および認識のタスクを行なってもらった．被験者は 22〜25 歳の大学，大学院生に行なってもらった．

子機は横に二つ並べ，左側を A，右側を B とする．各ジェスチャについては以下に説明する．

(a) A の上面に手を近づけるジェスチャ

(b) B の上面に手を近づけるジェスチャ

(c) B の手前の面に手を近づけるジェスチャ

(d) B の周囲を時計回りに一回転させるジェスチャ

(e) A から B へ手を移動させる (スワイプ) ジェスチャ

(f) B から A へ手を移動させる (スワイプ) ジェスチャ

認識するジェスチャには，似た MHI が得られると考えられるジェスチャを使用した．例えば，(a) の TopA では子機 A の上のセンサが反応した画像が得られると考えられる．一方で (f) の BtoA では (a) 同様に子機 A の上のセンサが反応するのに加えて，過去に子機 B を通った情報が含まれる MHI が得られると考えられる．

タスクとしてはシステムの各画面の使用方法の説明の後，先に述べた各ジェスチャを学習用に 5 回ずつ登録してもらい，その後各ジェスチャを 5 回ずつ行なった際の認識結果を記録した．

5.1.2　実験結果および考察

実験結果を表 5.1 に示す．表の値は被験者 4 名，各ジェスチャ5 回の計 20 回の認識結果の割合を示している．

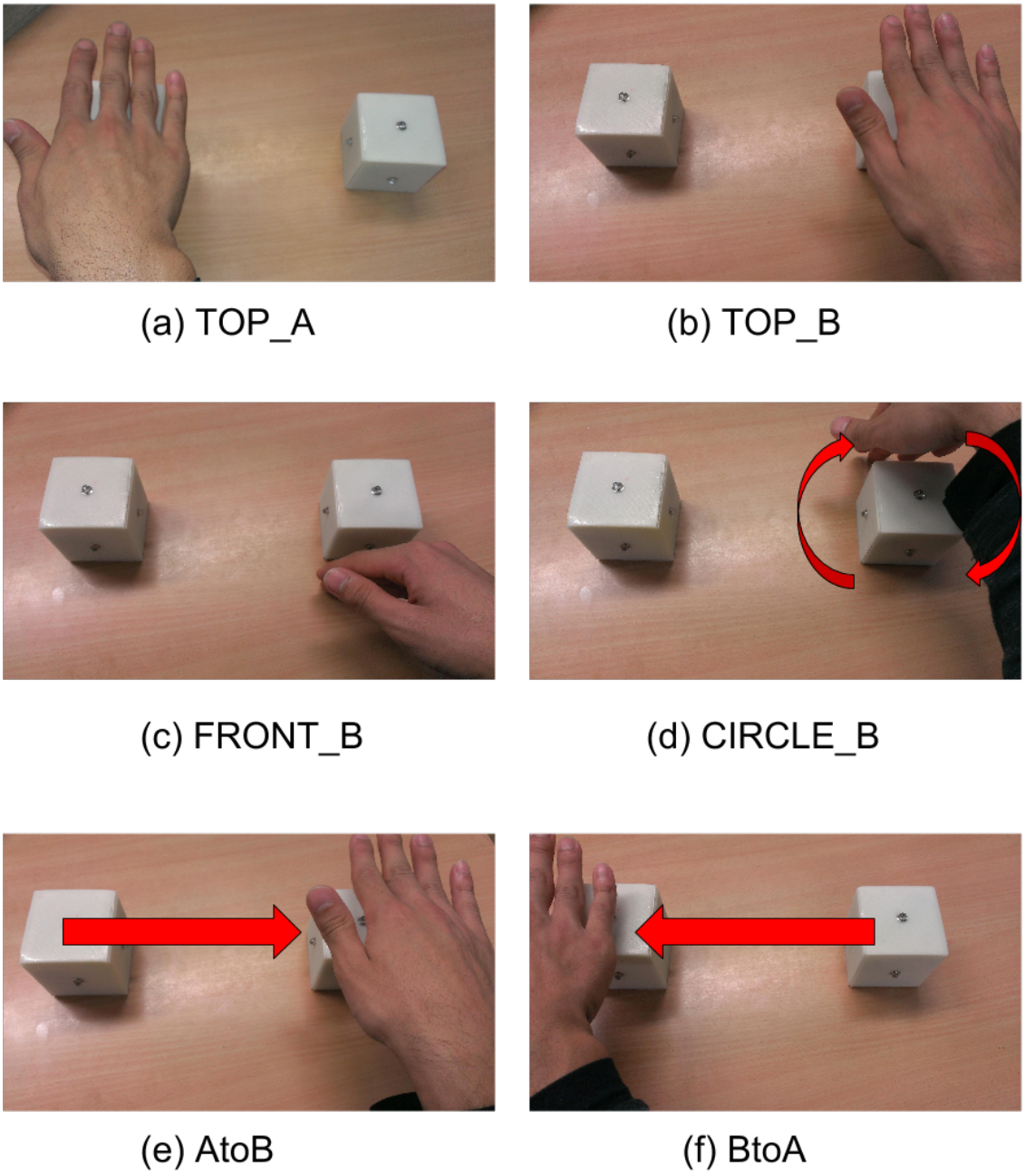

図 5.1: 実験 1 で学習，認識するジェスチャ

表 5.1: 実験 1 ジェスチャ認識結果

行なった ジェスチャ	認識されたジェスチャ						
	TopA	AtoB	TopB	FrontB	CircleB	BtoA	無反応
TopA	0.65	0.05	0	0	0	0.15	0.15
AtoB	0.05	0.75	0	0	0.05	0.15	0
TopB	0	0	0.95	0	0.05	0	0
FrontB	0	0.05	0	0.95	0	0	0
CircleB	0	0.05	0	0.05	0.9	0	0
BtoA	0	0.35	0	0.10	0	0.5	0.05

結果としては AtoB と BtoA のスワイプジェスチャにおいて誤認識が目立ったが，そのほかのジェスチャはほぼ誤認識なく認識することができた．

結果としては TopB，FrontB，CircleB では 9 割以上の認識することができた．一方で，TopA やスワイプのジェスチャでは認識率が悪い結果となった．

実験後，学習を行なった際のデータと認識時のジェスチャの様子を比べて誤認識の原因の考察を行なった．原因の一つとして，図 5.2 のように子機に対して少し手を奥に出すことで奥側面のセンサが大きく反応してしまい，学習時の手の位置と，実際にジェスチャを行なう際での手の微妙な差が，センサ値では大きくなり誤認識となっているケースが多く見受けられた．実際の誤認識の例を図 5.3 に示す．図 5.3 の上は学習時，下は認識時のデータであり，学習時は A の奥側面 (上中央のピクセル) が反応しているのに対して，認識時には手が内側に入ってしまい，センサが反応しなくなってしまい，その結果 CircleB と認識されてしまった．

もう一つの原因として，大半はセンサの近くでジェスチャを行なったが，学習時に図 5.4 のように子機に対して大きく上空でジェスチャを行なったため，MHI が薄くなり，センサに対して手を近づけていない状態が BtoA と誤認識されるケースも見られた．

一つ目の原因に関しては，今回作成した子機では LED が完全に外に露出するように取り付けていたため，側面の LED が広範囲の環境光の影響を受け，上からの光反応しすぎたことが問題と考えられる．そのため側面の LED は上から届く環境光の影響を小さくすることで認識精度を上げられると考えられる．二つ目の原因に関しては，システムの画像を見せる，センサが反応した際に LED を発光させることで提示するといった点に注目したもらった上で自由にシステムを使用し，あらかじめセンサが反応する範囲に慣れさせることで緩和できると考えられる．

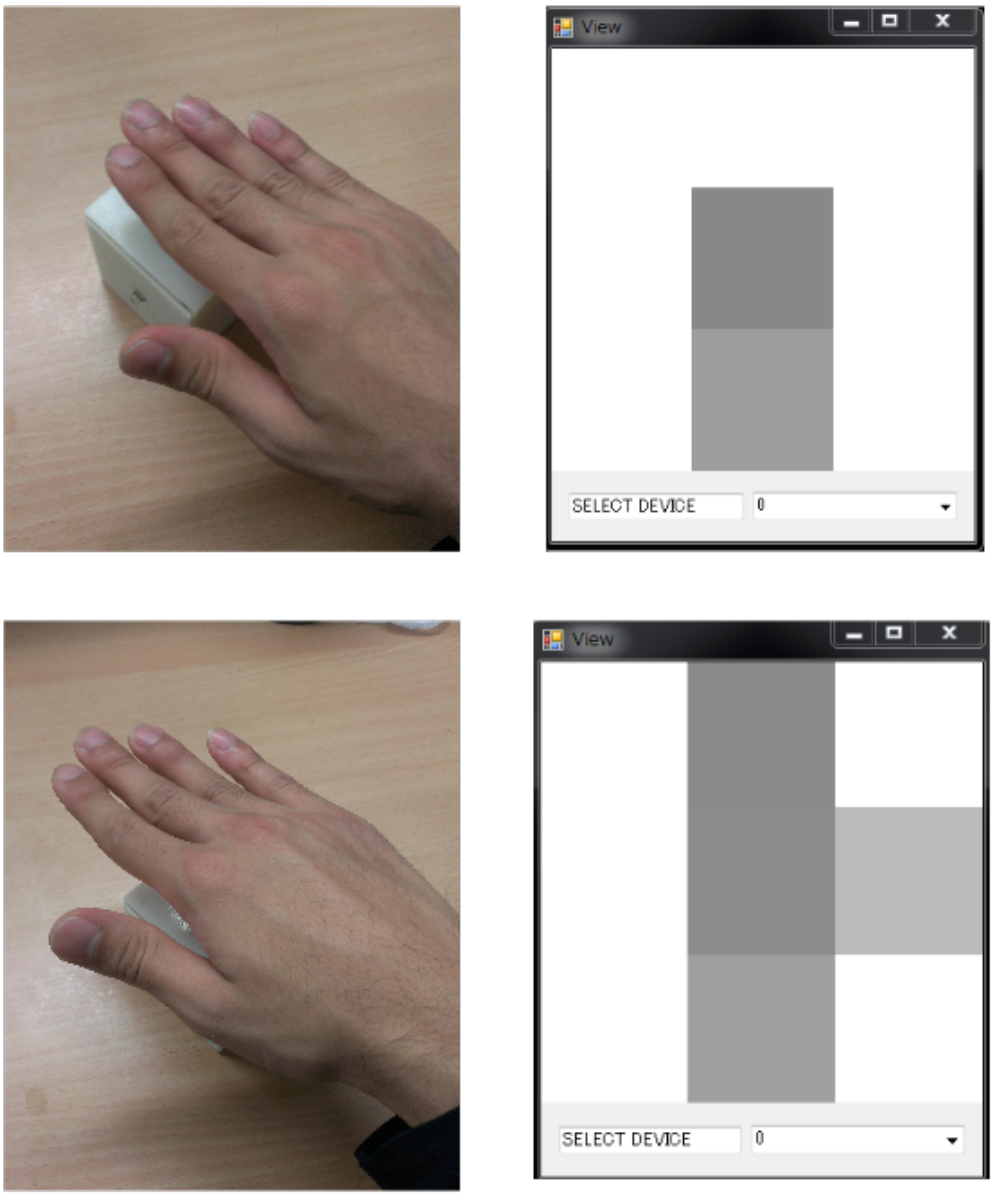

図 5.2: 手の位置による誤認識

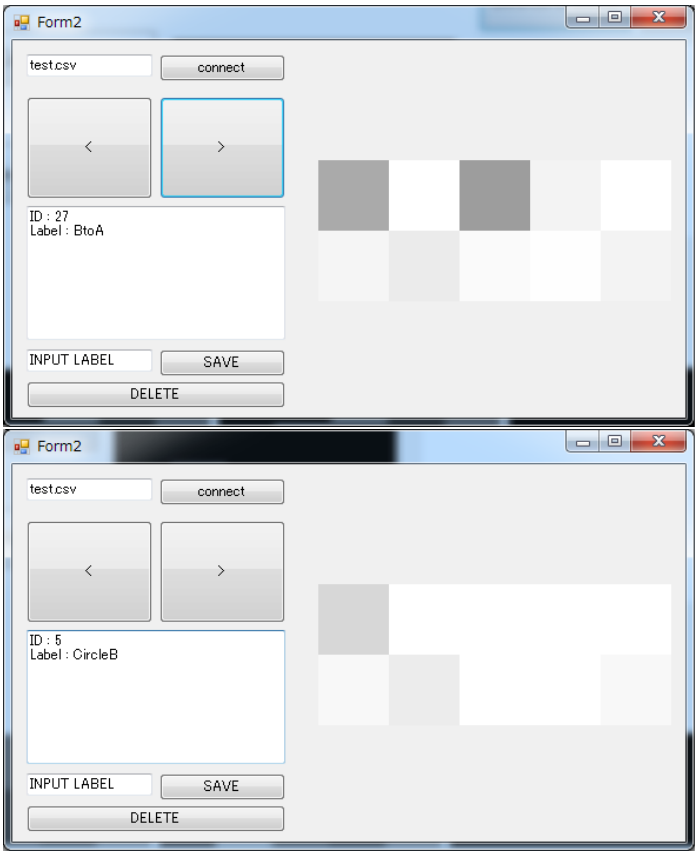

図 5.3: 手の位置の BtoA ジェスチャの誤認識例　(上:学習時　下: 認識時)

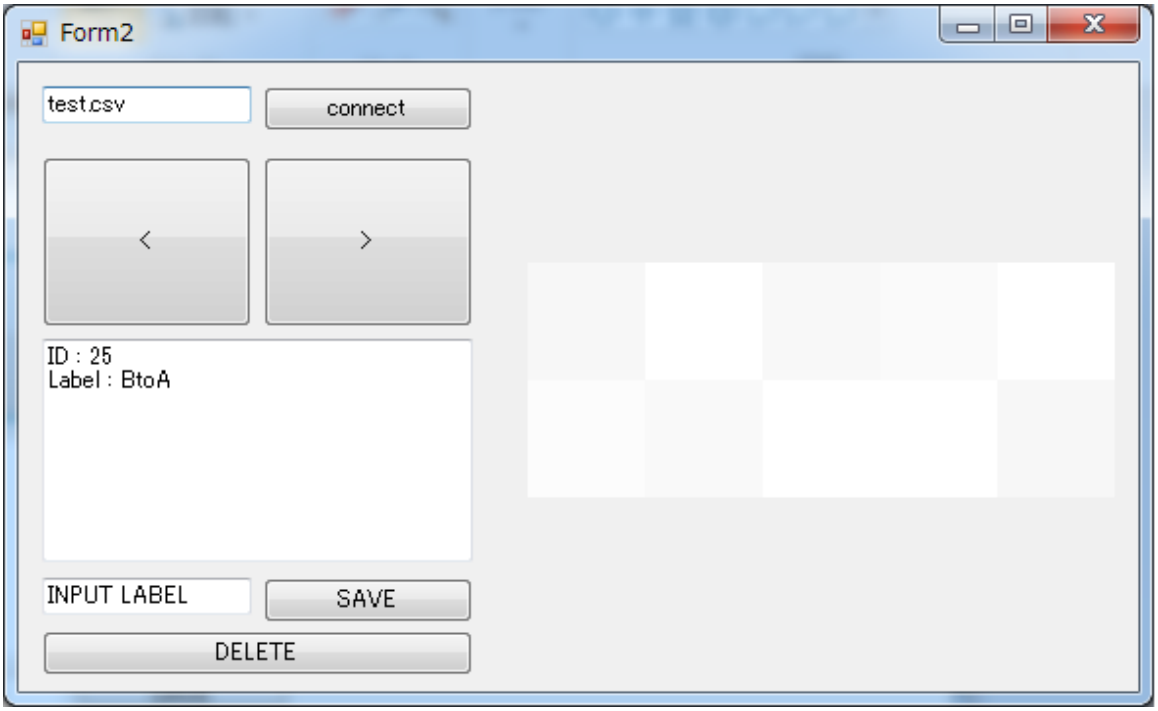

図 5.4: 手が高い位置でのジェスチャ登録

5.2 実験 2

実験 1 の結果から手の位置の原因以外の認識率を評価するため，再実験を行なった．実験内容と結果，および考察について以下に述べる．

5.2.1 実験内容

実験 1 の結果では，学習，認識時の手の位置による誤認識が多く見られた．そこで手の位置の原因を除いたシステムの認識率を評価するため，被験者にジェスチャを行なう際の手の位置を同じにするように伝えた上で再実験を行なった．また子機間での認識率を測るため実験 1 に加えて Front A，Circle A のジェスチャを追加した，8 つのジェスチャ登録，認識のタスクを行なった．実験 2 で認識するジェスチャを以下にまとめる．

(a) A の上面に手を近づけるジェスチャ

(b) A の手前の面に手を近づけるジェスチャ

(c) A の周囲を時計回りに一回転させるジェスチャ

(d) A から B へ手を移動させる (スワイプ) ジェスチャ

(e) B の上面に手を近づけるジェスチャ

(f) B の手前の面に手を近づけるジェスチャ

(g) B の周囲を時計回りに一回転させるジェスチャ

(h) B から A へ手を移動させる (スワイプ) ジェスチャ

被験者は実験 1 と同じく，22〜25 歳の大学，大学院生に行なってもらった．

5.2.2 実験結果及び考察

表 5.2: 実験 2 ジェスチャ認識結果

行なった ジェスチャ	認識されたジェスチャ								
	TopA	FrontA	CircleA	AtoB	TopB	FrontB	CircleB	BtoA	無反応
TopA	1	0	0	0	0	0	0	0	0
FrontA	0	0.95	0	0	0	0	0	0	0.05
CircleA	0	0	0.95	0	0	0	0	0.05	0
AtoB	0	0	0.25	0.7	0	0	0.05	0	0
TopB	0	0	0	0	1	0	0	0	0
FrontB	0	0	0	0	0	0.95	0.05	0	0
CircleB	0	0	0	0	0	0	1	0	0
BtoA	0	0.05	0.2	0.05	0	0	0.05	0.65	0

実験結果を表 5.2 に示す．被験者に手の位置を意識してもらうことで，TopA の認識率が実験 1 に比べて大幅に改善された．他にも Front A や Circle A，実験 1 でも高い認識ができていたジェスチャは 9 割以上認識することができた．一方で AtoB，BtoA のスワイプジェスチャはあまり認識率が改善されず，誤認識が残る結果となった．しかし，実験 1 ではスワイプが反対方向のジェスチャと誤認識されたのに対して，実験 2 ではどちらも追加したジェスチャである Circle A と誤認識される結果が多かった．そのため Circle A とスワイプジェスチャにおいて考察を行なった．CircleA では子機 A の側面を時計回りに回すため，得られる MHI としては子機 A の側面と正面のセンサが大きく反応しているものが得られる．これに対して BtoA のジェスチャでも最後に子機 A の側面を通り子機 A の上面と正面が大きく反応する MHI が得られる．そのため，これらが似た MHI となり誤認識されたと考えられる．またジェスチャ登録時の SAVE ボタンを押すタイミングを合わせることが難しいことから，AtoB でも子機 A が反応して子機 B が反応する前の MHI を登録してしまうケースが見られた．そのためこちらも Circle A と似た MHI が得られてしまったことが原因と考えられる．しかしスワイプジェスチャの誤認識に関しては，さらに原因を追究し改善する必要がある．

第6章　結論

本研究では，小型で連携可能な光センサ無線デバイスを用いて，ジェスチャの登録，認識を行なうことのできるシステムを提案した．本システムを利用することで，ユーザは認識したいジェスチャに対して必要なデバイスの個数や位置を容易に変更することができる．光センサ無線デバイスとして，立方体の形状で底面以外の 5 面に光センサとして LED を取り付けたデバイスを複数作成し，これらのセンサ値を用いてジェスチャの登録および認識を行うシステムを作成した．また実際にジェスチャの登録，認識のプロセスを行ってもらい，システムの評価を行なった．結果としては，ジェスチャの位置が少しずれることでセンサの反応が大きく変わる問題点や，ジェスチャを行なう速度に慣れが必要である問題点などがあり，認識精度には改善の必要があった．

今後は今回の結果を踏まえて，ジェスチャに適したデバイスや認識手法の考察を行い，認識率の改善を行なっていきたい．また本システムと他の IoT 製品を連携することで，認識したジェスチャを電子機器の操作に割り当てることができるように改良を行っていきたい．

謝辞

本論分を執筆するにあたり，指導教員の高橋伸准教授，そして田中二郎教授，志築文太郎准教授，三末和男教授，嵯峨智准教授，Simona Vasilache 助教にはゼミやミーティングを通して大変貴重なご意見をいただきました．特に高橋准教授には普段から普段から研究テーマ，研究の方向性，論文の執筆などに関して大変多くのご指導をいただきました．ここに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日常生活の中で様々な意見を頂きました．特にユビキタスチームの同期には，研究生活全般からそれ以外に渡っても多くの支えを頂き，励みになりました．ここに深く感謝いたします．最後に自分の生活を支えてくださった家族，日常生活や研究生活において支えてくださった皆様に深く感謝いたします．

参考文献

[1] Seokhwan Kim, Shin Takahashi, and Jiro Tanaka. Point-tap, tap-tap, and the effect of familiarity: to enhance the usability of see-and-select in smart space. *Information and Media Technologies*, Vol. 8, No. 1, pp. 97–108, 2013.

[2] 入江耕太, 若村直弘, 梅田和昇. ジェスチャ認識に基づくインテリジェントルームの構築. 日本機械学会論文集 C 編, Vol. 73, No. 725, pp. 258–265, 2007.

[3] Anusha Withana, Roshan Peiris, Nipuna Samarasekara, and Suranga Nanayakkara. zsense: Enabling shallow depth gesture recognition for greater input expressivity on smart wearables. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pp. 3661–3670, 2015.

[4] Stuart Taylor, Cem Keskin, Otmar Hilliges, Shahram Izadi, and John Helmes. Type-hover-swipe in 96 bytes: A motion sensing mechanical keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1695–1704, 2014.

[5] Daniel Wilson and Chris Atkeson. Simultaneous tracking and activity recognition (star) using many anonymous, binary sensors. In *Pervasive computing*, pp. 62–79. 2005.

[6] Christopher R Wren and Emmanuel Munguia Tapia. Toward scalable activity recognition for sensor networks. In *Location-and context-awareness*, pp. 168–185. 2006.

[7] 本田誠一, 福井健一, 森山甲一, 栗原聡, 沼尾正行. 赤外線センサーネットワークによる人物追跡. 人工知能学会全国大会論文集, No. 0, pp. 102–102, 2006.

[8] Jakub Segen and Senthil Kumar. Shadow gestures: 3d hand pose estimation using a single camera. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, Vol. 1. IEEE, 1999.

[9] Jeune Scott-Kemball. *Javanese shadow puppets: the Raffles collection in the British Museum*. British Museum, 1970.

[10] JL Raheja, Sishir Kalita, Pallab Jyoti Dutta, and Solanki Lovendra. A robust real time people tracking and counting incorporating shadow detection and removal. *International Journal of Computer Applications*, Vol. 46, No. 4, pp. 51–58, 2012.

[11] Nicholas Sheep Dalton. Taptiles: Led-based floor interaction. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces*, ITS '13, pp. 165–174, 2013.

[12] Junichi Akita. Interactive block device system with pattern drawing capability on matrix leds. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pp. 1059–1062, 2012.

[13] Florian Echtler, Thomas Pototschnig, and Gudrun Klinker. An led-based multitouch sensor for lcd screens. In *Proceedings of the fourth international conference on Tangible, embedded, and embodied interaction*, pp. 227–230, 2010.

[14] xbee-arduino. `https://github.com/andrewrapp/xbee-arduino`.

[15] Aaron Bobick and James Davis. The recognition of human movement using temporal templates. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 23, No. 3, pp. 257–267, 2001.

[16] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, Vol. 2, No. 3, pp. 27:1–27:27, May 2011.

[17] libsvm-net. `https://code.google.com/p/libsvm-net/`.