JAIST Repository
https://dspace.jaist.ac.jp/

Title	家庭内における統合位置情報システムに関する研究
Author(s)	塚越，航
Citation	
Issue Date	2017-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/14156
Rights	
Description	Supervisor:丹　康雄，情報科学研究科，修士

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修士論文

家庭内における統合位置情報システムに関する研究

1510036　　塚越 航

主指導教員　丹 康雄 教授
審査委員主査　丹 康雄 教授
審査委員　篠田 陽一 教授
審査委員　リム 勇仁 准教授

北陸先端科学技術大学院大学

情報科学研究科

平成 29 年 2 月

概 要

情報技術の発展により、世の中に存在する様々なモノに通信機能を持たせ、ネットワークに接続させて自動制御等を図る IoT という技術が注目されてきている。その一環として、家庭内の様々な家電や機器も高機能化が進み、ネットワーク接続ができる物が普及してきた。ネットワーク接続が可能となったことにより、家庭内の家電や機器は他の様々な機器と通信、連携し、様々な情報を取得、提供することが可能になり、その一環として使用者の位置を知ることも可能になる。また、家庭内には様々な年齢、性別や家庭内の立場が異なるユーザが複数存在し、それらユーザの状態に応じて家庭内の家電や機器はサービスを提供する必要がある事が、先行研究 [1] によって述べられている。そのため、位置情報を使用する家庭内の家電や機器は、ユーザの位置だけでなく、そのユーザが具体的に誰であるかというユーザの識別情報を知る必要もある。位置測位の手法としては GPS(Global Positioning System) がデファクトスタンダードとなってきているが、家庭内などの屋内に対しては GPS による測位が行えないという欠点がある。また屋内での場合、屋内の具体的にどこにいるかというような位置測定を行う必要があるため、屋内専用の位置測定手法を用いる必要がある。これまでに、屋内専用の位置測位手法は色々と考案されており、それらは大まかに、特定の機器を所持したユーザに対して、その機器に対する読み取り機が機器を検出することでユーザを検知する方法と、センサによりユーザを検知し位置を測位する方法の 2 つに分けられる。一般に、前者は機器を所持したユーザしか測位できないが、機器に ID が振られていることなどからユーザが誰かを容易に識別できる。後者ではユーザは機器を所持する必要はなく、全てのユーザに対し検出が可能となるが、ユーザの識別は難しくなる。本研究では、屋内での位置測位手法を複数用意し、それぞれにおいて検出したユーザの位置情報を収集、統合してより正確な一つのユーザの位置情報を導出できるような位置情報統合システム、かつそれにより得たユーザの位置情報を家庭内向けサービスプラットフォーム及びアプリケーション等に提供できるような技術の提案、実装を行う。具体的には、複数の屋内用の位置検出手法を屋内に設置し、設置した複数の位置検出手法が位置検出を行い、得た結果を位置情報統合システムに送る。位置情報統合システムは各位置検出手法から送信された位置検出結果を受け取り、それらのデータに対し統合処理を行い、一つの位置情報を導出する。その後位置情報統合システムは導出した位置情報を出力し、家庭内向けサービスプラットフォーム及びアプリケーション向けに提供することを目的とする。位置情報統合システムの設計として、利用する様々な位置検出手法と統合システム、統合システムと家庭内向けアプリケーションとの間にデータを変換するための API を設けた。また、位置情報統合システム内でのデータ構造として LocationData を設定し、屋内でユーザが大まかにどの区域にいるかを表すものとして Convexhull という概念を用い、データの統合にはこの Convexhull の和集合、差集合を取る形で実現した。また信頼性を上げるため、システムに対してユーザ推論を行う機能も実装した。システムの最終的な出力として、統合後のデータに応じ、6 段階の精度に分けて出力させるようにし

た。本研究では導入する位置検出手法として、RFID、人感センサ、カメラ型センサを設定し、これらからのデータの受け渡し方法も含め、統合システムの実装を行った。最後には実験として、利用する位置検出手法の数を変えた上での出力の比較、また設計したユーザ推論機能についての動作を見ることを目的に行った。その結果、LS を単独で用いるよりも複数の LS を用いた時の方が精度の高い出力が得られる事、またユーザ推論機能についても正しく動作している事が確認でき、屋内での位置検出手法を組み合わせる事でそれぞれの位置検出手法の欠点をカバーし、より信頼性の高い情報を得る事が可能となった。

目 次

第 1 章　はじめに **1**
1.1　研究背景 1
1.2　研究目的 2
1.3　本論文の構成 2

第 2 章　屋内位置検出手法 **4**
2.1　屋内位置検出手法の概要 4
2.2　屋内位置検出手法の関連技術 4
2.2.1　RFID (Radio Frequency IDentifier) 4
2.2.2　RSSI (Received Signal Strength Indication) 5
2.2.3　センサ 5
2.2.4　カメラ 5

第 3 章　提案する位置情報統合システム (ILS) **7**
3.1　概要 7
3.2　全体図 7
3.3　位置情報データ構造 8
3.3.1　ConvexHull 8
3.3.2　LocationData 11
3.4　LS (Location System) 13
3.5　API_LS (LS-ILS 間) 14
3.6　統合 (Integrate) クラス (モジュール) 14
3.6.1　ユーザ推論 15
3.7　API_SERVICE (ILS-Service Platform 間) 17
3.8　データの変換 17

第 4 章　使用する LS **18**
4.1　Speedway 18
4.2　人感センサ 18
4.3　カメラ型センサ 21

第 5 章　実装 **23**
5.1　開発環境 23
5.2　LS 23
5.2.1　Speedway 23
5.2.2　人感センサ 23
5.2.3　カメラ型センサ 24
5.3　Convexhull 24
5.3.1　Convexhull 同士の和集合 24
5.3.2　Convexhull 同士の差集合 26
5.4　部屋の設定ファイル 27
5.5　ILS 全体 29
5.5.1　API_LS 29
5.5.2　統合 (Integrate) クラス 29
5.5.3　API_SERVICE 30

第 6 章　実験と評価 **31**
6.1　実験方法 31
6.2　実験結果 38
6.2.1　実験１・２ 38
6.2.2　実験３ 40
6.3　評価・考察 45

第 7 章　まとめ **47**

第 8 章　今後の課題 **48**

図 目 次

3.1 ILS 全体図 9
3.2 3 次元空間上の Convexhull 10
3.3 Convexhull での直方体の制作 10
3.4 球状の Convexhull 11
3.5 円錐状の Convexhull 12
3.6 ユーザ識別ができている Convexhull 12
3.7 LocationData 13
3.8 ユーザ識別誤認例 16
3.9 ユーザ識別確率導入図 16

4.1 Speedway 19
4.2 Speedway で使用するアンテナ 19
4.3 Speedway で使用する RFID タグ 20
4.4 人感センサ 20
4.5 カメラ型センサ 21
4.6 動体検知の様子 22
4.7 速度表示の様子 22

5.1 Convexhull の和集合を取った様子 26
5.2 Convexhull の差集合を取った様子 27

6.1 4F 輪講室及び LS の配置図 32
6.2 実験１、２を行う上でのユーザの経路図 36
6.3 実験３を行う上でのユーザの経路図 37
6.4 実験１、２　実験の様子 38
6.5 実験１、２　出力データ精度レベル 39
6.6 実験１　LS のユーザ検知の様子 (Speedway のみ) 39
6.7 実験２　LS のユーザ検知の様子 40
6.8 実験３　実験の様子 41
6.9 実験３　位置検出の様子１ 41
6.10 実験３　位置検出の様子２ 42
6.11 実験３　位置検出の様子３ 43

6.12 実験３　位置検出の様子４ 43

表 目 次

3.1　API_SERVICE　出力データ精度レベル分け 17

第1章　はじめに

本章では研究背景と研究目的，本論文の構成を示す.

1.1　研究背景

情報技術の発展により、世の中に存在する様々なモノに通信機能を持たせ、ネットワークに接続させて自動制御等を図る IoT という技術が注目されてきている。その一環として、家庭内の様々な家電や機器も高機能化が進み、ネットワーク接続ができる物が普及してきた。ネットワーク接続が可能となったことにより、家庭内の家電や機器は他の様々な機器と通信し、様々な情報を取得、提供することが可能になり、その一環として使用者（以下、ユーザと呼ぶ）の位置を知ることも可能になる。家庭内の家電や機器にとってそれを使用する側であるユーザの位置は重要であり、家電はユーザの位置を知ることで様々なサービスを自動で提供することができ、省エネにも役立たせることができる。

また、家庭内には様々な年齢、性別や家庭内での立場が異なるユーザが複数存在し、それらユーザの状態に応じて家庭内の家電や機器はサービスを提供する必要がある事が、先行研究 [1] によって述べられている。そのため、ユーザの位置情報を利用する家庭内の家電や機器は、ユーザの位置だけでなく、そのユーザが具体的に誰であるかというユーザの識別情報を知る必要もある。

位置測位の手法としては GPS(Global Positioning System) がデファクトスタンダードとなってきている。しかし、家庭内などの屋内に対しては GPS による測位が行えないという欠点がある。また屋内での場合、屋内の具体的にどこにいるかというような位置測定を行う必要があるため、屋内専用の位置測定手法を用いる必要がある。

これまでに、屋内専用の位置測位手法は色々と考案されてきている。それらは大まかに、特定の機器を所持したユーザに対して、その機器に対する読み取り機が機器を検出することでユーザを検知する方法と、センサによりユーザを検知し位置を測位する方法の 2 つに分けられる。

一般に、前者は機器を所持したユーザしか測位できないが、機器に ID が振られていることなどからユーザが誰かを容易に識別できる。後者ではユーザは機器を所持する必要はなく、全てのユーザに対し検出が可能となるが、ユーザの識別は難しくなる。

1.2　研究目的

これまでに提案・開発されてきた屋内での位置検出手法には利点もあるが欠点もあり、単体からは得られる情報も少ないものが多くある。そのため、各々の位置検出手法で得られた位置情報をまとめ、より信頼性の高い位置情報の導出を行う技術、またネットワーク接続ができる家庭内の家電や機器の普及に伴い、導出したユーザの位置情報をそれらに提供できるような技術についても検討が求められる。

本研究では、屋内においてその様々な屋内の位置測位手法により収集した情報を統合し、誰がどこにいるかという正確なユーザの位置情報を導き出すような技術の提案、実装を行う。まずは、その実現のために導入可能な位置測位技術についての検討を行う。その後、作成するシステムの仕様を計画し、実装を行う。作成したシステムに対し実験・評価を行い、有用性などについて課題を検討する。

1.3　本論文の構成

本論文は以下の構成となっている.

- 第 1 章
  - 研究の背景と目的，本論文の構成を示す。
- 第 2 章
  - 本研究に関連、また導入を検討する、屋内位置測定技術についてを説明する。
- 第 3 章
  - 本研究で提案する位置情報統合システムについての仕様を説明する。
- 第 4 章
  - 本研究で提案する位置情報統合システム内で使用する、位置情報データのデータ構造についてを説明する。
- 第 5 章
  - 位置情報統合システムの実装についてを述べる。
- 第 6 章
  - 位置情報統合システムの実験・評価についてを述べる。
- 第 7 章

  - 本研究におけるまとめを述べる。

- 第 8 章
  - 今後の課題についてを述べる。

第2章　屋内位置検出手法

本章では，屋内での位置検出、またそれに関連する技術についてを記述する．

2.1　屋内位置検出手法の概要

位置検出の手段としては、現在 GPS がデファクトスタンダードとなってきているが、家庭内などの屋内に対しては屋内の具体的にどの辺りにいるかという情報を知る必要があるため、一般的に GPS による位置測位が行えなく、屋内専用の位置検出手法を用いる必要がある。これまでに、様々な屋内でのユーザの位置検出手法が提案、開発されてきている。それらは大まかに、ユーザが特定の機器を所持し、それに対する読み取り機により位置検出を行う方法と、屋内に設置したセンサによって位置検出を行う方法に分けられる。これらの手法により特定ユーザの位置を知るためには、

- 読み取り機またはセンサの屋内での具体的な位置
- ユーザの識別情報
- 読み取り機またはセンサとユーザとの距離
- ユーザ検出の有無

が必要である。

2.2　屋内位置検出手法の関連技術

屋内位置検出に関する関連技術を説明する。

2.2.1　RFID (Radio Frequency IDentifier)

RFID(Radio Frequency IDentifier)[2] とは、電波 (電磁波) を使って物や人を識別する技術である。一般には、識別・検知したい個体に ID 情報を搭載したタグを持たせ、タグに対する読み取り機がタグを読み取ることで、タグを持たせている個体を検出、またタグの ID 情報を読み取ることで個体が識別可能となる。利用する電波の周波数により、計測

可能な範囲、また計測の精度が異なってくる。RFID は大きく分けて、電池を内蔵しないパッシブタグと、タグに電池を内蔵するアクティブタグの２種類に分けられる。パッシブタグは読み取り機からの電波を受信し、その電波を反射し読み取り機に送り返す事で検知を行う。この反射した電波に ID などの情報を乗せることで ID も読み取れるが、一般的にこの反射波は微弱であるため、計測できる距離は短い。アクティブタグは電池を必要とするが、その分電波を発するためにパッシブタグに比べ長い距離で計測が可能となる。

2.2.2 RSSI (Received Signal Strength Indication)

RSSI (Received Signal Strength Indication)[3] とは受信電波の強度の事で、位置の推定にこの値の強弱が用いられる。ユーザに RSSI が測定可能な機器を所持させる事で、発信機から発せられた電波を受信し、その RSSI を計測する事で発信機との距離を推定する。利用例としては、Beacon と呼ばれる発信機を設置し、スマートフォンなどの機器を所持したユーザが Beacon からの電波を検知し、RSSI を計測する事で特定の Beacon の近くにいる事を検知する。この情報を利用しているアプリケーションも存在する。また、Wi-Fi などの無線 LAN に対しても利用例としてあり、スマートフォンなどの受信機を所持したユーザが無線 LAN のアクセスポイントからの RSSI を計測する事で位置検出を行う事例もある。

2.2.3 センサ

センサによる屋内での位置測定 [4] は前述までのものとは違い、一般にユーザは機器を所持する必要はなく、機器の所持にかかわらず全てのユーザに対し検知を行えるという利点がある。しかし、一般的にユーザの識別は行えない。ユーザ検出に用いられるセンサに関しては大きく分けて、アクティブセンサとパッシブセンサの 2 種類に分けられる。

アクティブセンサは、センサ側から赤外線や光などのエネルギーを発し、検知対象の個体がそれを遮る事でユーザの位置検出を行うセンサである。

パッシブセンサは、センサの検知範囲内に検知対象の個体が入った時に、個体が発する赤外線、また熱や熱による温度変化などを感知し、それによりユーザの位置検出を行うセンサである。

2.2.4 カメラ

屋内位置測位におけるカメラ型センサは、屋内にカメラを設置し、それにより写し出された画像を用いた位置検出を行う。位置検出に関しては、主に動体検知機能によりユーザを検知する。動体検知機能とは、カメラの撮影範囲内に動いている物体を検知すると作動し、検知しない時は待機状態になる機能である。カメラが動いている物体（ユーザ）を検知する時は作動するので、この状態でいる時はユーザを検知しているとみなす事ができ

る。また現在では、カメラと被写体（ユーザ）との距離を測定する機能 [5] などといった研究も行われている。

第3章　提案する位置情報統合システム (ILS)

本章では、本研究で提案する位置情報統合システムについてを記述する.

3.1　概要

本研究では、前章で示したような、屋内での位置測位手法を複数用意し、それぞれにおいて検出したユーザの位置情報を収集、統合してより正確な一つのユーザの位置情報を導出できるような位置情報統合システム (Integrated Location System:ILS, 以下、ILS と呼ぶ)、かつそれにより得たユーザの位置情報を家庭内向けサービスプラットフォーム及びアプリケーション等に提供できるような技術の提案、実装を行う。具体的には、複数の屋内用の位置検出手法 (以下、LS(Location System) と呼ぶ) を屋内に設置し、設置した複数の LS が位置検出を行い、得た結果を ILS に送る。ILS は各 LS から送信された位置検出結果を受け取り、それらのデータに対し統合処理を行い、ユーザの位置情報を導出する。その後 ILS は導出した位置情報を出力し、家庭内向けサービスプラットフォーム及びアプリケーションに提供する。この一連の動作を繰り返し行い、一定時間ごとに毎回ユーザの位置情報を出力する。また統合処理において、ILS がそれまでの処理において認識できたユーザを記憶しておく機能も有しておき、その後の統合処理において認識できたユーザの位置が不明になった時にでも屋内での位置を推定する機能を持たせる。この機能により、特定のユーザに対して正確な位置情報が導出できないときでも、位置推定を行うことで（正確な情報を導出したときと比べて精度はある程度落ちたものになるが）位置情報を提供できる事が可能となり、特定のユーザに対してトラッキングを行う必要があるアプリケーションに対し有用となる。

本研究では、ユーザの位置測位は３次元空間上で行うこととし、屋内の各部屋ごとに独立した座標系を持つものとして扱う。

3.2　全体図

ILS の全体図を図 3.1 に示す。図 3.1 において、各 LS と ILS の間、ILS と家庭内向けサービスプラットフォーム及びアプリケーションの間には API を設け、ここで各 LS、ILS、サー

ビスプラットフォームなどで扱うデータ形式に適応するようにデータ構造の変換を行う。ここで、各 LS と ILS との間に設ける API を API_LS、ILS と家庭内向けサービスプラットフォーム及びアプリケーションの間に設ける API_SERVICE とし、以下、この名称で呼ぶ。

全体の流れについて、まず設置した複数の LS(図 3.1 中,$LS_1, LS_2, ...LS_n$) は位置検出を行い、結果を ILS に送信する。ILS はその結果を、API_LS を介して受信する。API_LS により、データ形式が異なる各 LS からの位置情報データが、全て ILS 内で扱うデータ形式に変換される。ILS 内で扱う位置情報のデータ形式については、3.3 節で説明する。その後、ILS 内でのデータ形式に変換された各 LS からのデータに対し統合処理を行う。統合処理は、ILS 内部にある統合クラス (モジュール) が行う。各 LS からの位置情報を統合クラスに入力し統合処理を行い、一つの位置情報を導出し、出力する。その後、その導出した位置情報を API_SERVICE を介して、家庭内向けサービスプラットフォーム及びアプリケーションに送り出す動作を行う。

3.3 位置情報データ構造

ここでは、ILS で扱う位置情報のデータ構造についてを述べる。

3.3.1 ConvexHull

本研究では、屋内でユーザが大まかにどの区域にいるかを表す指標として、凸包 (Convexhull, 以下、Convexhull と呼ぶ)[6] という概念を用いる。３次元空間上における Convexhull とは、３次元空間における n 点の集合 P において、P の点を頂点として持ち、かつ P の点を全て内部に含む最小の凸多面体の事である。３次元空間上における Convexhull の例を図 3.2 に示す。図 3.2 は、３次元空間上においてランダムに 100 個配置した点に対し制作した Convexhull を示している。100 個の点を点集合 P とおくと、P の点は全て Convexhull の頂点または Convexhull の内部に存在することになる。

図 3.2 はランダムに配置した点に対し Convehull を制作したものであるが、点をランダムではなく全て指定して配置する事で、任意の立体図形を作成することができる。例として、直方体を制作したいと考えた時、点を制作したい直方体の頂点と同じ位置に与えて Convexhull を制作すると、作成された Convexhull は直方体になる。(図 3.3)

本研究では、LS による検出範囲をこの Convexhull で表す。LS とユーザとの推定距離が解る LS の検出範囲は、LS の位置を中心とし、その推定距離を半径とする球状の Convexhull で表し、検出したユーザはこの内部にいるものとみなす。球状の Convexhull の例を図 3.4 に示す。

また、センサ類の LS の検出範囲は、LS の位置を頂点とする円錐状の Convexhull で表す。センサ類の LS に対しては、ユーザ検出の情報も使用し、ユーザを検出したときユー

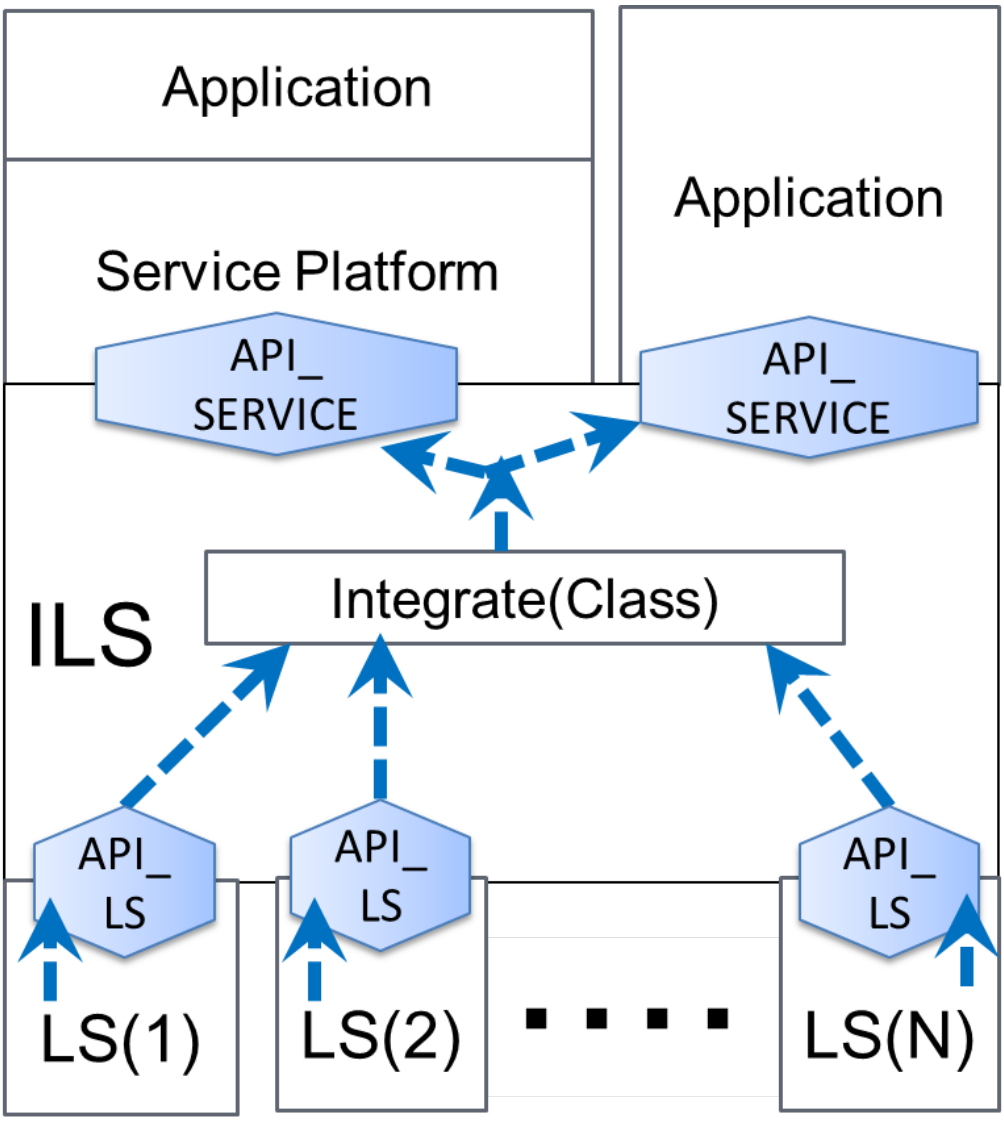

Application
Application
Service Platform
API_
SERVICE
API_
SERVICE
ILS
Integrate(Class)
API_
LS
API_
LS
API_
LS
LS(1)
LS(2)
LS(N)

図 3.1: ILS 全体図

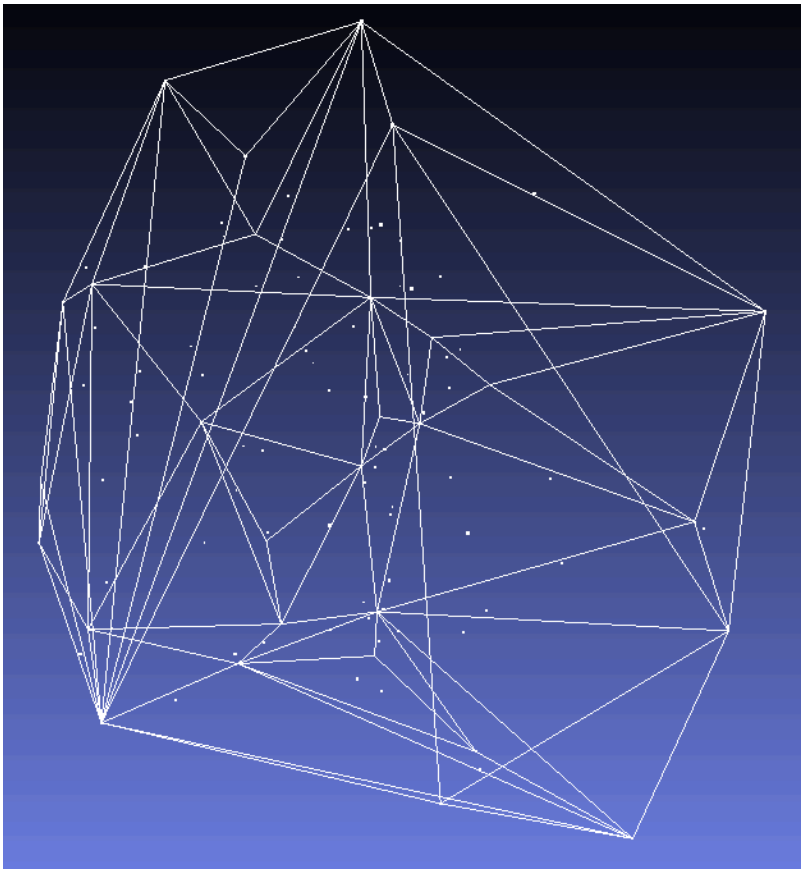


図 3.2: 3 次元空間上の Convexhull

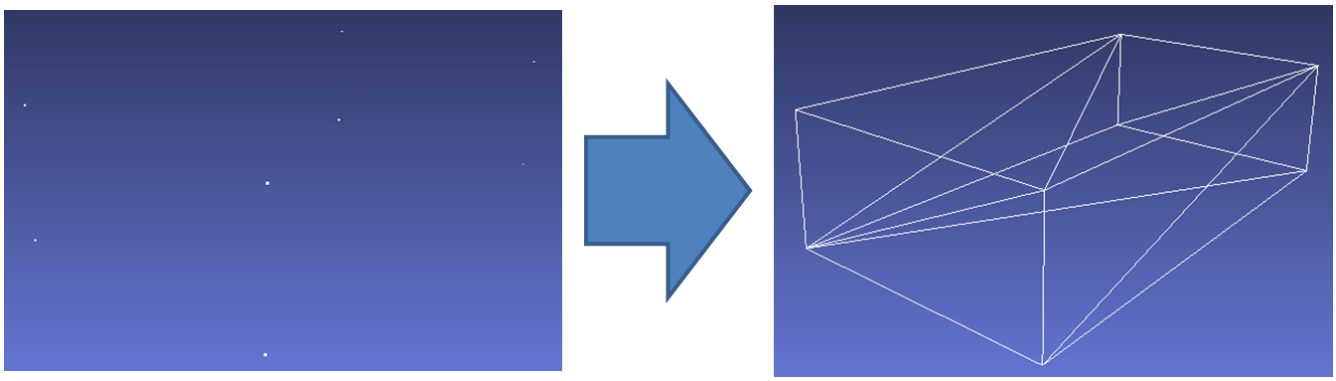


図 3.3: Convexhull での直方体の制作

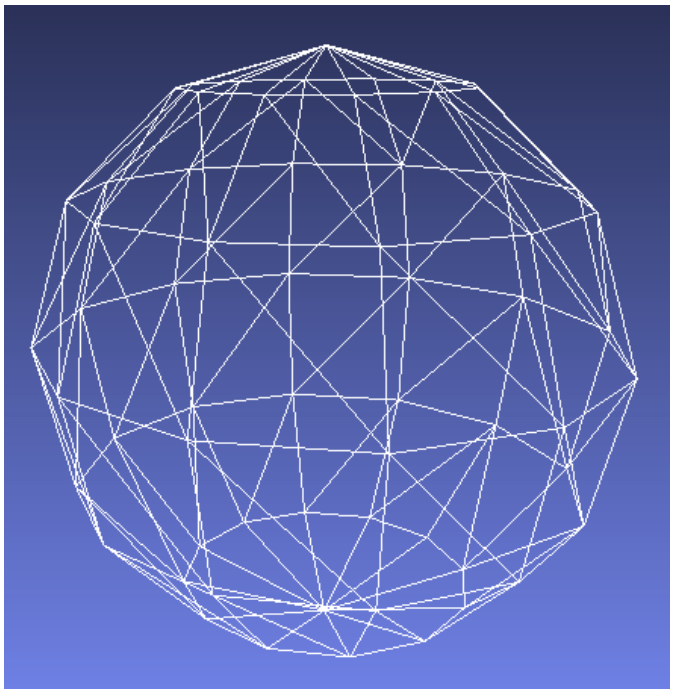


図 3.4: 球状の Convexhull

ザはこの範囲内にいるものとみなし、ユーザを検出しなかった時はこの範囲内にユーザは一人もいないものとみなす。円錐状の Convexhull の例を図 3.5 に示す。

また、ユーザ識別ができる LS については、Convexhull に色をつけることでそのユーザであることを表す。ユーザ識別ができている Convexhull の例として、図 3.6 を示す。図 3.6 では、識別したユーザを赤色で示している。

3.3.2 LocationData

次に、ILS 内で扱う位置情報データ形式として、LocationData を説明する。LocationData は ILS 内で扱う位置情報データを表すクラスである。LocationData クラスは図 3.7 に示した要素を持ち、ILS の中で各ユーザの位置情報を示すデータとしてやり取りされる。これらの要素はそれぞれ、以下の意味を持つ。

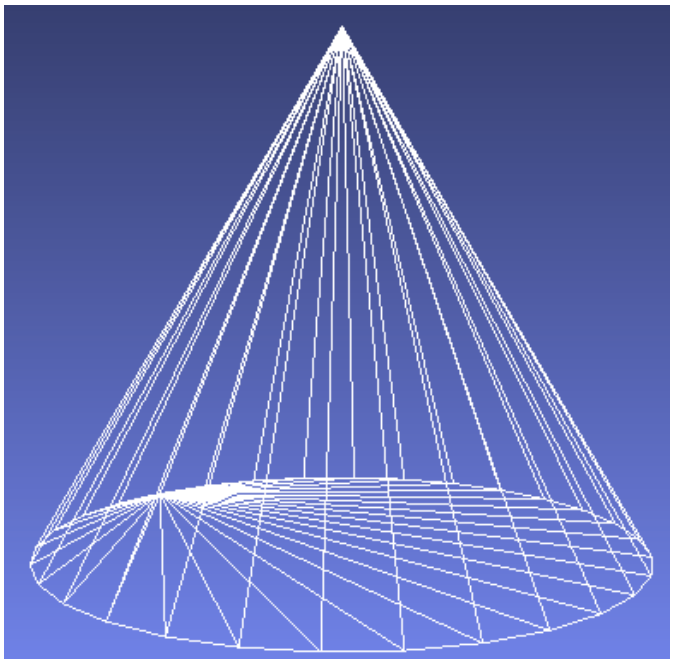


図 3.5: 円錐状の Convexhull

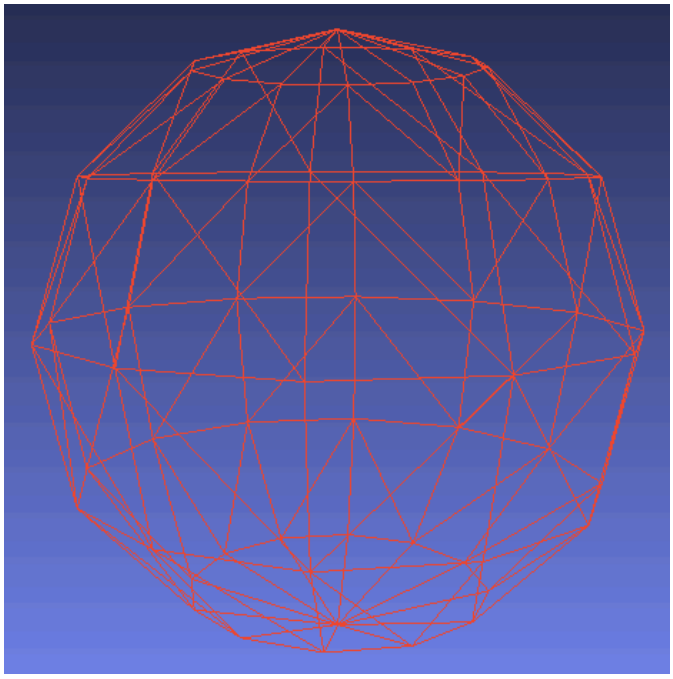


図 3.6: ユーザ識別ができている Convexhull

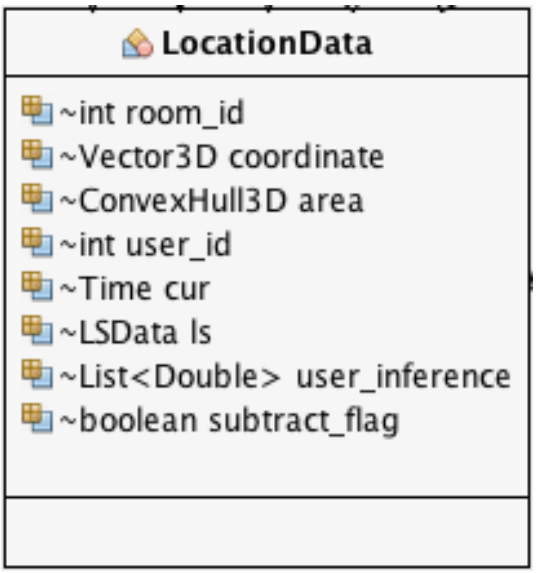


図 3.7: LocationData

・room_id　：部屋 ID
・coordinate　：ユーザの座標
・area　：ユーザがいうる区域 (Convexhull)
・user_id　：ユーザ ID
・cur　：データ取得時刻
・ls　：LS の情報 (位置や角度など)
・user_inference　：ユーザ推論値
・subtract_flag　：area(Convexhull) を和集合 (false) するか差集合 (true) させるか

3.4 LS (Location System)

LS(Location System) は前述の通り、屋内用の位置検出手法の事を指し、ILS に位置情報データを送るために設置する。LS は位置検出により得たデータを ILS に送るが、その際には第 2 章で述べた情報を含んだ、以下の情報を ILS に送るようにする。

- LS の位置
- ユーザの識別情報 (ユーザ識別できる LS のみ)
- LS とユーザとの距離 (ユーザとの距離が推定できる LS のみ)
- ユーザ検出の有無
- センサの角度、検出範囲 (センサ類のみ)

- Convexhull のタイプ（球, 円錐）

ユーザ識別ができない LS においては、ユーザの識別情報はユーザ識別情報不明 (ユーザ ID:-1) として送信する。また、LS とユーザとの距離が推定できない LS においては、LS とユーザとの距離は 0 として送信する。センサ類に関しては、センサが向いている角度、またそのセンサが検出できる範囲についての情報も加えて送信する。また、LS に応じ、その LS から得れる情報からユーザのいうるエリアを表現するために、3.3.1 節で述べた球、円錐のどの Convexhull を使うかという情報も送信する。

本研究では導入した LS(第 4 章で記載) の関係上、ユーザの位置を推定するための情報として、LS は ILS に対しこれらの情報を送るように設定している。しかし、LS にはこれら以外の情報を出力できる物や、ユーザの位置を推定するためにこれら以外の情報を用いる物もあり、また今後そのような LS が登場する場合もある。そのため、今回導入した LS 以外の LS を今後新たに導入し、かつこれら以外にユーザの位置情報を表現する要素を要する場合は、これら LS から ILS に送る要素を適宜変更・追加する必要がある。

3.5 API_LS (LS-ILS 間)

API_LS は ILS において、設置した各 LS と ILS との間に設ける API である。各 LS が出力した位置情報データを ILS 内で扱う位置情報データ形式である LocationData に変換する作業を行う。具体的には、API_LS は各 LS からのデータ (3.4 節で述べた要素) を入力とし、そのデータを元に Convexhull、LocationData を作成する。API_LS は出力として、この LocationData を出力する。API_LS は各 LS から送られた情報を受信する。各 LS から送られる位置情報データの形式は 3.4 節で示した通りである。ILS に新たに LS を導入するときは LS からこの情報を送らせるように設定する。しかし 3.4 節と同様に、新たに導入する LS の特性上、3.4 節で設定した LS から ILS に送る情報の他に、別の新しい情報を新たに追加するとき、API_LS もその追加した情報を受信するように設定する必要がある。また、それに加えて API_LS が LS からのデータを LocationData に変換する作業にも変更が生じるため、その動作も適宜変更する必要がある。

また、ILS は API_LS で各 LS からのデータを得るが、API_LS でユーザ識別ができているデータを得た時、それまでに識別できたユーザのリストを作成する。このユーザリストは、統合処理や API_SERVICE での出力時に、各ユーザの位置情報を導出、またユーザの推論 (3.6.1 節) を行う時に利用される。

3.6 統合 (Integrate) クラス (モジュール)

統合 (Integrate) クラス（モジュール）は ILS 内でデータの統合を行うクラスであり、以下、統合クラスと呼ぶ。API_LS により出力された LS の数だけある LocationData を入力

とし、内部で統合の処理を行い、その結果として統合された LocationData を出力する。ここで、出力する LocationData は部屋内で識別できたユーザごとにそれぞれ出力する。

そのため出力する LocationData の個数は、最高で識別できたユーザの数である。識別できたユーザが誰であるか、または何人いるかは、API_LS で作成したユーザリストを参照して行う。勿論、入力データの中には識別したユーザ全員のデータが無い場合もあるので、ユーザが特定できないデータに対しては識別できなかったユーザ（以下、不明ユーザと呼ぶ）として扱い、出力する。

具体的な統合の処理は、LocationData に含まれている Convexhull を利用し、Convexhull の和集合や差集合を取る形でより正確なユーザのいるエリアの導出を行う。統合処理を行う前に部屋全体を示す Convexhull の作成を行い、その Convexhull に対して LocationData に含まれている Convexhull を利用し統合を行う。API_LS では、センサ類の LS に関してはユーザの検知の有無にかかわらずセンサの検出範囲を示す Convexhull を生成し、ユーザ検知の有無などといった情報とともに LocationData を生成するが、この時ユーザ検知が有った Convexhull に対しては和集合を、ユーザ検知が無かった Convexhull には差集合を取らせるようにする。Convexhull に和集合か差集合を取らせるかの判別は、LocationData の subtract_flag を参照して判断する。

3.6.1 ユーザ推論

統合クラスにおいて、ユーザ推論を行う機能を持たせる。ILS はユーザの位置導出に加え、ユーザの識別も行うが、屋内での複数のユーザの状況、場合により ILS が一度断定した識別情報が異なってしまうこともある。例として、図 3.8 のようなパターンが発生した場合、ILS が誤った識別を行う事もあり、導出した位置情報を提供する家庭内向けサービスプラットフォームやアプリケーションに対し悪影響を及ぼす恐れもある。そのため、ユーザ識別においては、ユーザ識別における信頼度の指標として、確率的な指標を位置情報データのデータ構造に組み入れる必要がある。ユーザ識別において、確率的な指標を組み入れた図を図 3.9 に示す。確率的な指標を導入させた上でトラッキングし、途中あるユーザが識別できた時に別の不明ユーザの確率を計算する事によって、不明ユーザに対する識別の信頼度を向上させる。

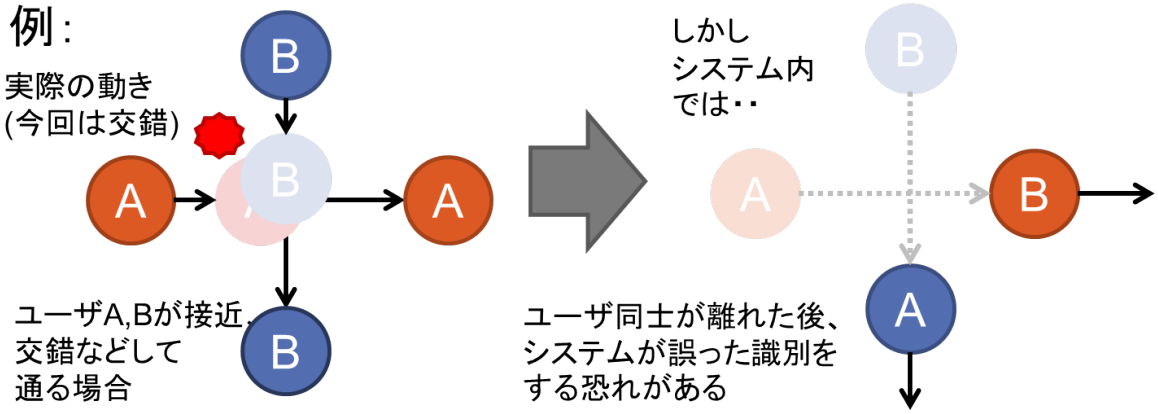


図 3.8: ユーザ識別誤認例

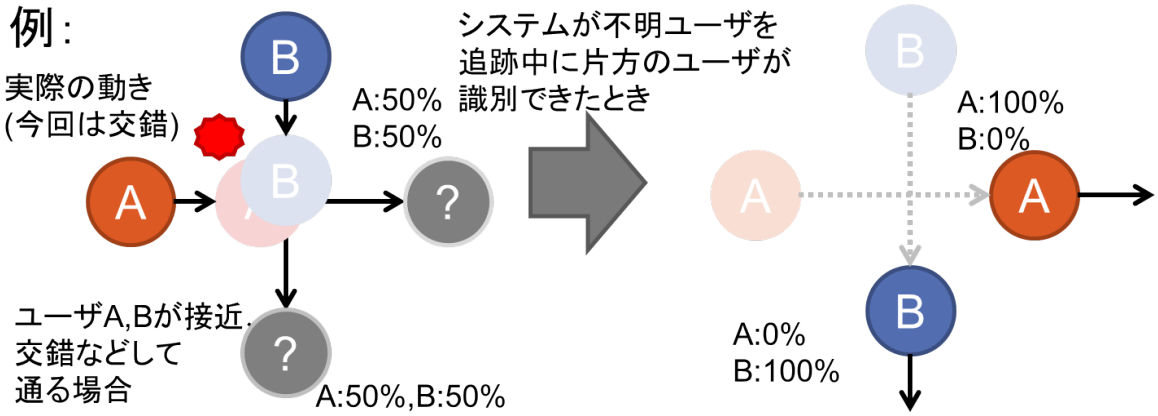


図 3.9: ユーザ識別確率導入図

3.7 API_SERVICE (ILS-Service Platform 間)

API_SERVICE は ILS において、ILS と家庭内向けサービスプラットフォーム及びアプリケーションとの間に設ける API であり、ILS 内の統合クラスが出力した各ユーザ (ILS がそれまでに認識しているユーザで、その時点での統合で導出できなかったユーザは不明ユーザとする) における位置情報データを入力とし、出力として ILS が認識している各ユーザの位置情報を出力する。ここで、API_SERVICE が出力する各ユーザの位置情報は、入力となる統合クラスによる各ユーザの位置情報の精度によりレベル分けする。本研究では API_SERVICE が出力する位置情報データの構造、精度を、表 3.1 のように規定した。

精度レベル	データ構造	説明
1	(x,y,z,room)	ユーザがどの部屋 (room) で、どの座標 (x,y,z) にいるか
2	(area,room)	ユーザがどの部屋 (room) で、どの区域 (area) にいるか
3	(room)	ユーザがどの部屋 (room) にいるか
4	[(room),(room),,,]	ユーザがいうる部屋 (room) を列挙し、リストで表す
5	(true/false)	ユーザが家庭内にいるかいないかを返す (true/false)
6	unknown	ユーザが検知できない、不明とする

表 3.1: API_SERVICE　出力データ精度レベル分け

3.8 データの変換

ここまでに述べたデータの変換においてを説明する。まず、LS が位置検出を行った結果 (3.4 節のユーザ識別情報、距離、検知など) を出力し、データを ILS に送る。ILS の API_LS がこの LS からの情報に応じて屋内でのユーザがいうるエリアを表す Convexhull、LocationData を作成する。その LocationData を、統合クラスに渡し、統合を行う。その結果出力された各ユーザの LocationData は、その中身に応じて API_SERVICE により出力される。API_SERVICE により出力されるデータの構造は、表 3.1 の通りである。

第4章　使用するLS

本章では、今回のILSの実装で使用するLSについてを説明する。今回のILSの実装において、用いたLSは以下の3つである。

- Speedway(RSSI・RFID)
- 人感センサ
- カメラ型センサ

以下、この各LSについての操作、データの送信方法についてを説明する。

4.1　Speedway

SpeedwayとはImpinj社[7]の製品で、RFIDの機器である(図4.1)。アンテナ(図4.2)を4つまで接続でき、アンテナから電波を発する事で、専用のRFIDタグ(図4.3)を検知し、読み取ったRFIDタグのIDを得る。このSpeedwayで使用するRFIDタグは電池を使わない、パッシブ型のRFIDタグである。また、SpeedwayにはRFIDタグのRSSIを測定する機能もあり、これにより検知したRFIDタグのRSSIも知る事ができ、RFIDタグとの距離も推定できる。Impinj社はSpeedwayの操作に関するAPI、サンプルコード(Java)[8]を公開しており、今回の実装ではそちらを一部利用している。

4.2　人感センサ

本研究では、人感センサは焦電型赤外線センサを用いる。これは、検出範囲内において人体が放出する赤外線を感知することでユーザを検知するパッシブ型のセンサである。検出範囲は120度、検出距離は7mと規定されている。今回はこの人感センサをRaspberry Piに接続し、対応するGPIO番号の出力を読み取る形で人感センサの出力を読み取る形で使用を行う。(図4.4)

図 4.1: Speedway

図 4.2: Speedway で使用するアンテナ

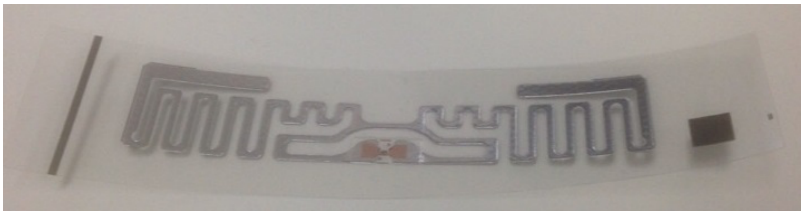


図 4.3: Speedway で使用する RFID タグ

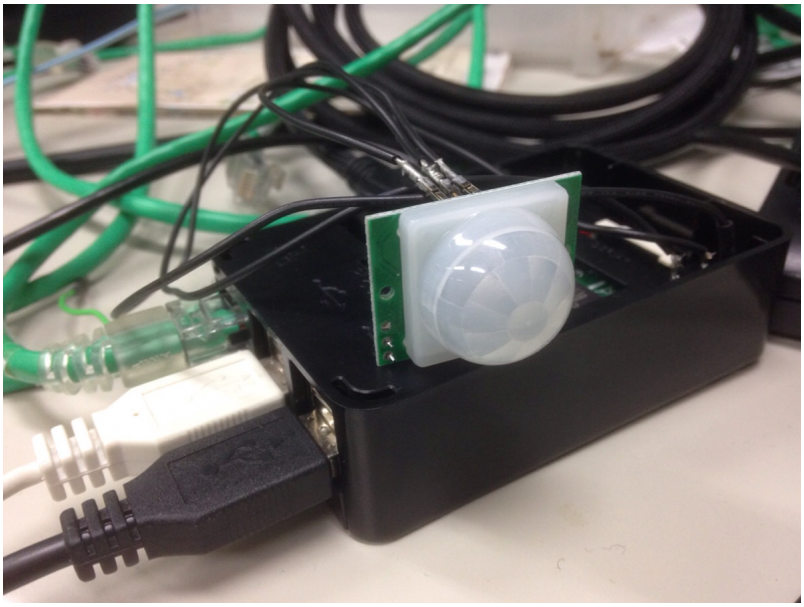


図 4.4: 人感センサ

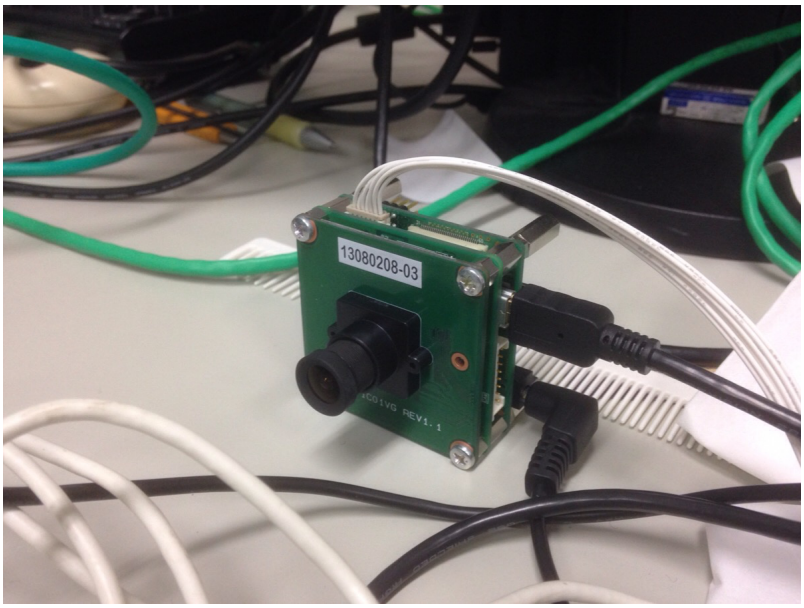


図 4.5: カメラ型センサ

4.3 カメラ型センサ

カメラ型センサは、SMK 社 [9] から提供して頂いたカメラ型センサを利用する。外観図を図 4.5 に示す。

このカメラ型センサは、画面を 9 分割し、それぞれの画面内において動いている物体を検知した時にのみ画面を表示し、検知してない時は表示しない機能を有する。その様子を図 4.6 に示す。図 4.6 において、 9 分割されている画面のうち、画面が映っている 7 つの画面については動いている物体を検知しているということを表しており、画面が表示されてない 2 つの画面については、動いている物体を検知していないことを表している。

また、9 分割された画面それぞれにおいて、検知した物体の速度を出力する機能も有している。その様子を図 4.7 に示す。

図 4.6: 動体検知の様子

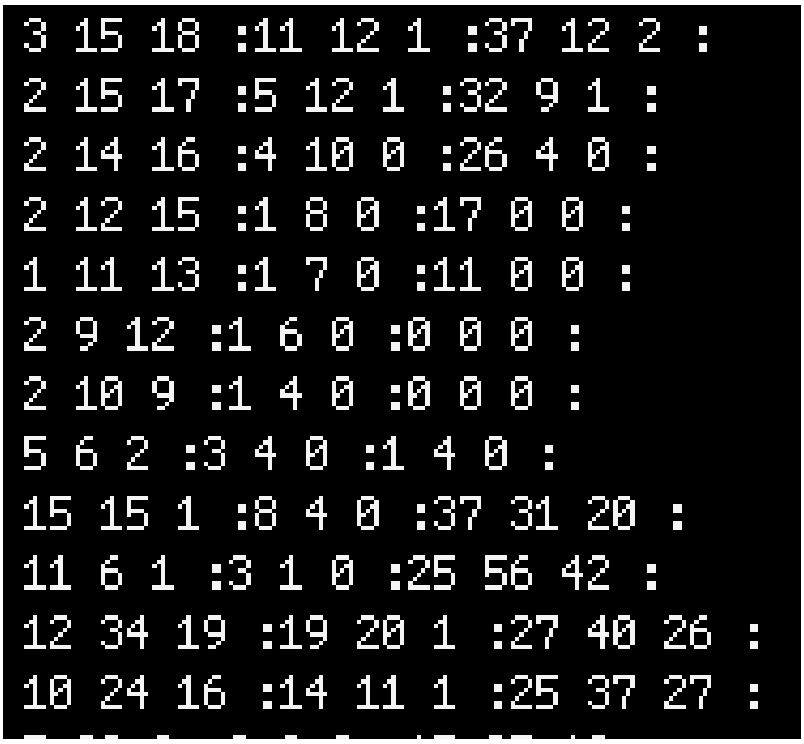


図 4.7: 速度表示の様子

第5章　実装

本章では、ILS の実装についてを述べる。

5.1　開発環境

今回の ILS の実装において用いた開発環境は以下の通りである。

- 開発環境：Mac OS X El Capitan
- 開発言語：Java
- 統合開発環境：NetBeans
- Raspberry Pi 3 Model B

5.2　LS

各 LS についての実装を述べる。

5.2.1　Speedway

Speedway においては、4.1 節で述べた Impinj 社が公開している Java プログラムを利用した。また、これらは NetBeans でも利用できる。

5.2.2　人感センサ

人感センサにおいては、Raspberry Pi に接続して使用する。Raspberry Pi の GPIO 部分と人感センサをつなぎ、その後は Raspberry Pi 側で対応する GPIO 番号の出力を見るように操作する。Raspberry Pi は得た人感センサの出力を ILS 側に送るが、送信の手段としては、ソケットを用いたネットワーク通信を利用する。そのため、Raspberry Pi 側であらかじめサーバプログラムを動作させておき、ILS 側がクライアントプログラムを動作し Raspberry Pi と通信することで人感センサの出力を得る。

5.2.3 カメラ型センサ

カメラ型センサにおいても、Raspberry Pi に接続する形で使用する。こちらは、9 画面それぞれで検知した物体の速度を数字で出力する機能があるので、その出力を利用することでカメラの範囲内にユーザがいるかどうかを判断する。出力された数字が 0 なら動体検知が無いのでユーザがいないと判断、0 より大きい場合には動体検知があるという事になるのでユーザがいると判断できる。これにより、ユーザ検知の有無を ILS に送り出す。ILS への送信方法としては、人感センサの時と同様にソケットを用いたネットワーク通信を利用する。こちらも Raspberry Pi 側であらかじめサーバプログラムを動作させておき、ILS 側がクライアントプログラムを動作し Raspberry Pi と通信することでカメラ型センサの出力を得る。

5.3 Convexhull

次に、Convexhull の実装についてを述べる。本研究では、3 次元空間上における Convexhull を表現する要素として、

- 頂点
- 辺
- 面 (ファセット)

を規定する。これらはそれぞれ、リストで管理する。3 次元空間上における Convexhull を作成するアルゴリズム [6] を参考に、3 次元空間上における点集合を入力とし、その点集合に対する Convexhull を作成する java プログラムの作成を行った。3 次元空間上における Convexhull を作成するアルゴリズムを Algorithm 1 に示す。また、上記に述べた頂点、辺、面のデータ構造としては、Apache Commons Math ライブラリ [10] を使用し表現した。最終的に、計算が行われ全ての点、辺、ファセットが入った Convexhull が出力されるが、出力された Convexhull を可視化するために、Convexhull を.ply ファイルに書き落とす機能もつけ、Convexhull の可視化についても実装を行った。本研究では作成した.ply ファイルの可視化ツールとして、Meshlab[11] を用いた。

5.3.1 Convexhull 同士の和集合

統合処理で 2 つの Convexhull の共通部分を算出するために、2 つの Convexhull の共通部分を算出して 1 つの Convexhull を導出するプログラムの実装を行った。5.3 節で実装した Convexhull を作成する java プログラムは、頂点の集合を入力すればそれらに対する Convexhull を作成してくれるため、共通部分の Convexhull の作成にはこのプログラムを用いて行う。2 つの Convexhull を A,B とおき、A と B における共通部分 $(A \cap B)$ を 1 つ

Algorithm 1 Calculate Convexhull(3D)

Require: 3次元空間における n 点の集合 P

Ensure: P の Convexhull:CH(P)

```
４面体を構成する P の４点 p_1, p_2, p_3, p_4 を求める。
C ← CH(p_1, p_2, p_3, p_4)
残りの点のランダムな順列 p_5, p_6, ..., p_n を求める。
全ての可視点対 (p_t,f) を求めて衝突グラフ g を初期化する。ただし、f は C のファセット (面) であり、t > 4 である。
for r ← 5 to n do
  {(*p_r を C に挿入する:*)}
  if F_conflict(p_r) は空でない {(∗すなわち、p_r は C の外にある∗)} then
    F_conflict(p_r) のファセットを全て C から削除する
    p_r から見える領域の境界を順に辿り (その領域は F_conflict(p_r) のファセットから構成されている) 地平面の辺のリスト L を順に作る
    for すべての e ∈ L do
      三角形のファセット f を作ることにより e を p_r につなぐ
      if f が e に沿った近傍のファセットと同一平面にある then
        f と f' を１つのファセットに統合する。ただし、その衝突リストは f' のものと同じである。
      else
        {(*f の衝突リストを定める:*)}
        g に f に対する節点を作る。
        f_1 と f_2 を元の Convexhull で e に接していたファセットとする。
        P(e) ← P_conflict(f_1) ∪ P_conflict(f_2)
        for 全ての点 p ∈ P(e) do
          f が p から見えるなら、(p,f) を g に加える。
        end for
      end if
    end for
    p_r に対応する節点と F_conflict(p_r) のファセットに対応する接点をそれらに接続する枝とともに g から削除する。
  end if
end for
return C
```

の Convexhull と捉えた時、この Convexhull の頂点は A の頂点のうち B の内部にあるもの、A の辺と B のファセットの交点、同様に B の頂点のうち A の内部にあるもの、B の辺と A のファセットの交点で構成される。なので、２つの Convexhull を入力とし、これらの頂点を計算し出力した後、この頂点を入力とし 5.3 節で作成した java プログラムで計算を行わせれば、２つの Convexhull A,B における共通部分 $(A \cap B)$ をの Convexhull が導出される。2 つの Convexhull A,B において、和集合をとった Convexhull を図 5.1 に示す。

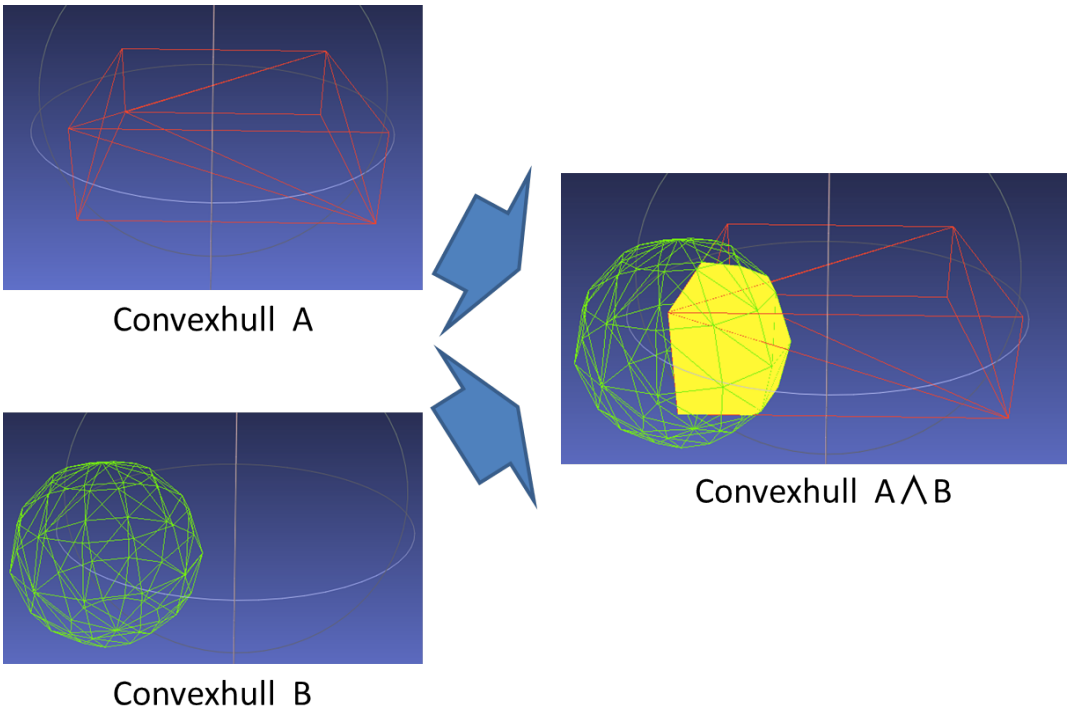


図 5.1: Convexhull の和集合を取った様子

5.3.2 Convexhull 同士の差集合

同様に統合処理で２つの Convexhull の差集合を算出する必要がある。これにおいても 5.3 節で実装した java プログラムを用いて行う。２つの Convexhull を A,B において、A から B の部分を引いた $(A \setminus B)$Convexhull の頂点は、A の頂点のうち B の内部にないもの、A の辺と B のファセットの交点、B の頂点のうち A の内部にあるもの、B の辺と A のファセットの交点で構成される。そのため同様に２つの Convexhull を入力とし、これらの頂点を計算し出力した後、この頂点を入力とし 5.3 節で作成した java プログラムで計算を行わせれば、２つの Convexhull A,B における差集合 $(A \setminus B)$ の Convexhull が導出される。2 つの Convexhull A,B において、差集合をとった Convexhull を図 5.2 に示す。図 5.2 は、Convexhull A(赤い球状の Convexhull) から Convexhull B(緑色の球状の Convexhull) の差

集合を取った Convexhull $(A \setminus B)$(黄色の Convexhull) を示している。ただし、差集合においては凹面を持つものも存在するが、差集合の立体を構成する頂点を与えて Convexhull を計算した場合、Convexhull の定義上、凹面の部分が埋め合わされて凸面となった立体となるため、差集合の Convexhull においては、厳密な差集合を作成できるとは限らない場合も存在する。

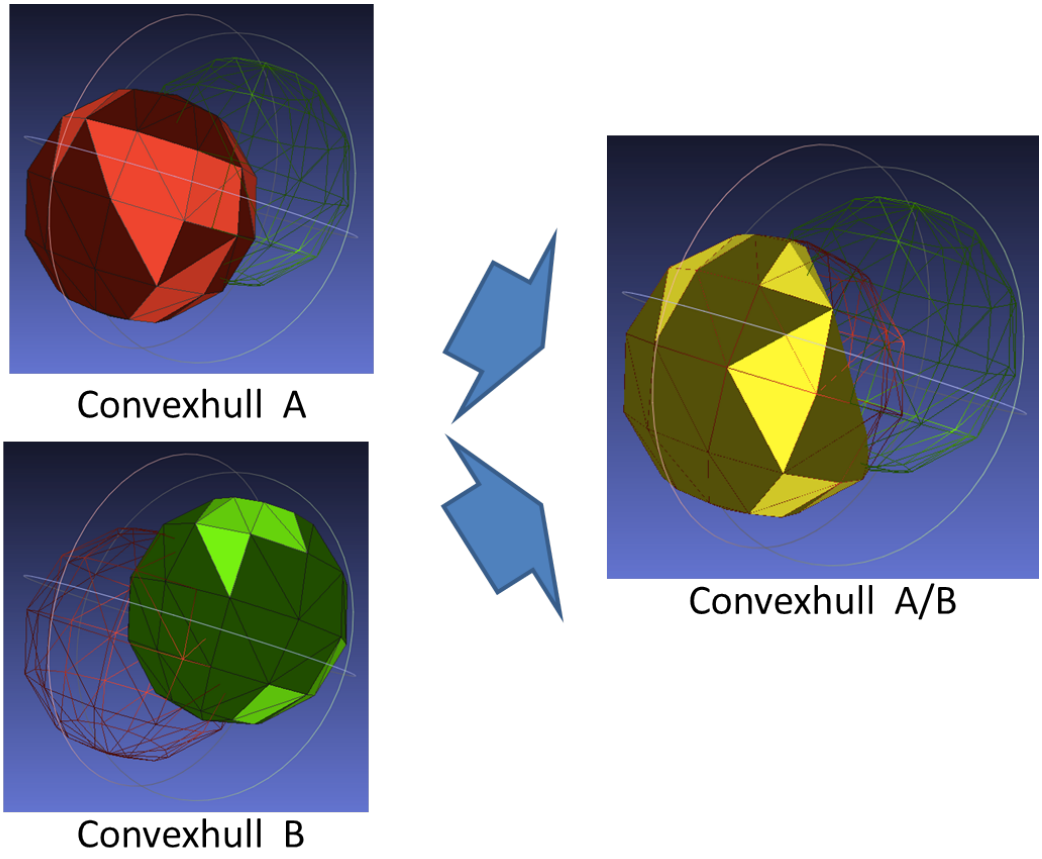


図 5.2: Convexhull の差集合を取った様子

5.4 部屋の設定ファイル

ILS に部屋の情報を知らせるために、部屋の情報を記した設定ファイルを作成し、ILS 起動時に読み込ませて取得を行う。部屋の情報とは、部屋の名前 (ID)、幅、奥行き、高さ、また部屋内に設置している主な LS についての情報がある。ILS は起動時に設定ファイルからこのような部屋の情報を読み込み、部屋を示す Convexhull の作成を行う。部屋の設定ファイルは JSON 形式で作成する。設定ファイルの例をコード 5.1 に記載する。

コード 5.1: 部屋の設定ファイル例 (json 形式)

```
{
    "Room":[
        {
            "name":"RoomA",
            "Room_id":1,
            "floor":4,
            "width":576,
            "depth":1026,
            "height":270,
            "LS":[

                {
                    "name":"Speedway",
                    "LS_id":1,
                    "coordinate":[
                        {
                                        "x":1,
                                        "y":480,
                                        "z":100
                        },

                        {
                                "x":370,
                                "y":1,
                                "z":85
                        }
                    ]
                },

                {
                    "name":"HumanSensor",
                    "LS_id":2,
                    "coordinate":[
                        {
                                "x":370,
                                "y":270,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":330,
                                "y_angle":90
                        },

                        {
                                "x":370,
                                "y":270,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":30,
                                "y_angle":90
```

```
                        }
                    ]
                }
            ]
        }
    ]
}
```

コード 5.1 において、最初に部屋の情報”Room”の配列形式で記載する。部屋ごとの具体的な情報として、部屋の名前 (”name”), 部屋 ID(”Room_id”), 部屋がある階 (”floor”), 部屋の幅 (”width”), 部屋の奥行き (”depth”), 部屋の高さ (”height”), 部屋内に設置されている LS の情報 (”LS”) を記載する。更に部屋内に設置されている LS については、LS の名前、部屋内での具体的な位置、センサ類においては検知範囲を示す直円錐の高さ (”height”), 直円錐の軸（頂点と底面の円の中心を通る直線）と母線とをなす角度 (”top_angle”), また３次元空間において直円錐を、直円錐の頂点を通る y 軸に平行な直線を回転軸として回転させる角度 (”x_angle”), 同様に直円錐の頂点を通る x 軸に平行な直線を回転軸として回転させる角度 (”y_angle”) といった情報を記載している。本研究では円錐をこれらの情報を用いて制作する。LS については以上のような、検出範囲を示すための具体的な情報も記載する。

5.5 ILS 全体

ILS に関する実装を述べる。

5.5.1 API_LS

API_LS は設置する各 LS ごとにインスタンスを生成する形で行う。また、LS と API_LS との間にバッファを設け、まず各 LS は出力結果をこのバッファに格納する。その後、API_LS がこのバッファから LS からの出力結果を取り出して ILS 側に持ってくる。その後、適した Convexhull を作成する。出力としては、LocationDatan のリストになる。

5.5.2 統合 (Integrate) クラス

統合クラスにおいては、APi_LS で生成した LocationData のリストを入力とし、LocationData に含まれている Convexhull に対して和集合または差集合をとる操作を行わせることでユーザの位置導出を行う。統合の処理の前に、部屋の設定ファイルから部屋を表す直方体状の Convexhull を作成する。その後、この Convexhull を用いて LocationData との統合処理を行っていく。ただ、入力の Convexhull の中には ILS が認識しているユーザの識別情報を持つ Convexhull が無い場合もあるので、その場合は不明ユーザの Convexhull

にユーザ推論の確率的指標を付加させて表示させる。最終的な出力には、最高でILSが識別できたユーザの数のLocationDataを出力する。

5.5.3 API_SERVICE

APi_SERVICEはILSが統合の結果出力した各ユーザの位置情報(LocationData)に応じ、ILSが認識しているユーザの位置情報を出力する。ただし、LocationDataに含まれる情報により出力する各ユーザの位置情報の精度を変える必要がある。出力するデータは表3.1の通りである。

第6章　実験と評価

実装したILS, また設置したLSを評価するために実験を行った。

6.1　実験方法

実験場所は4F輪講室で行い、Speedway、人感センサ、カメラ型センサを設置し、実験を行った。

また、今回の実験では以下の事項を確認することを目的とした。

- 1つのLSのみの場合と複数のLSで統合した場合との結果を比較する
- ユーザ推論の過程を見る

実験場所とした4F輪講室、及び設置したLSの場所を示す図を図6.1に記載する。

また、今回の実験で使用した部屋の設定ファイルをコード6.1に示す。

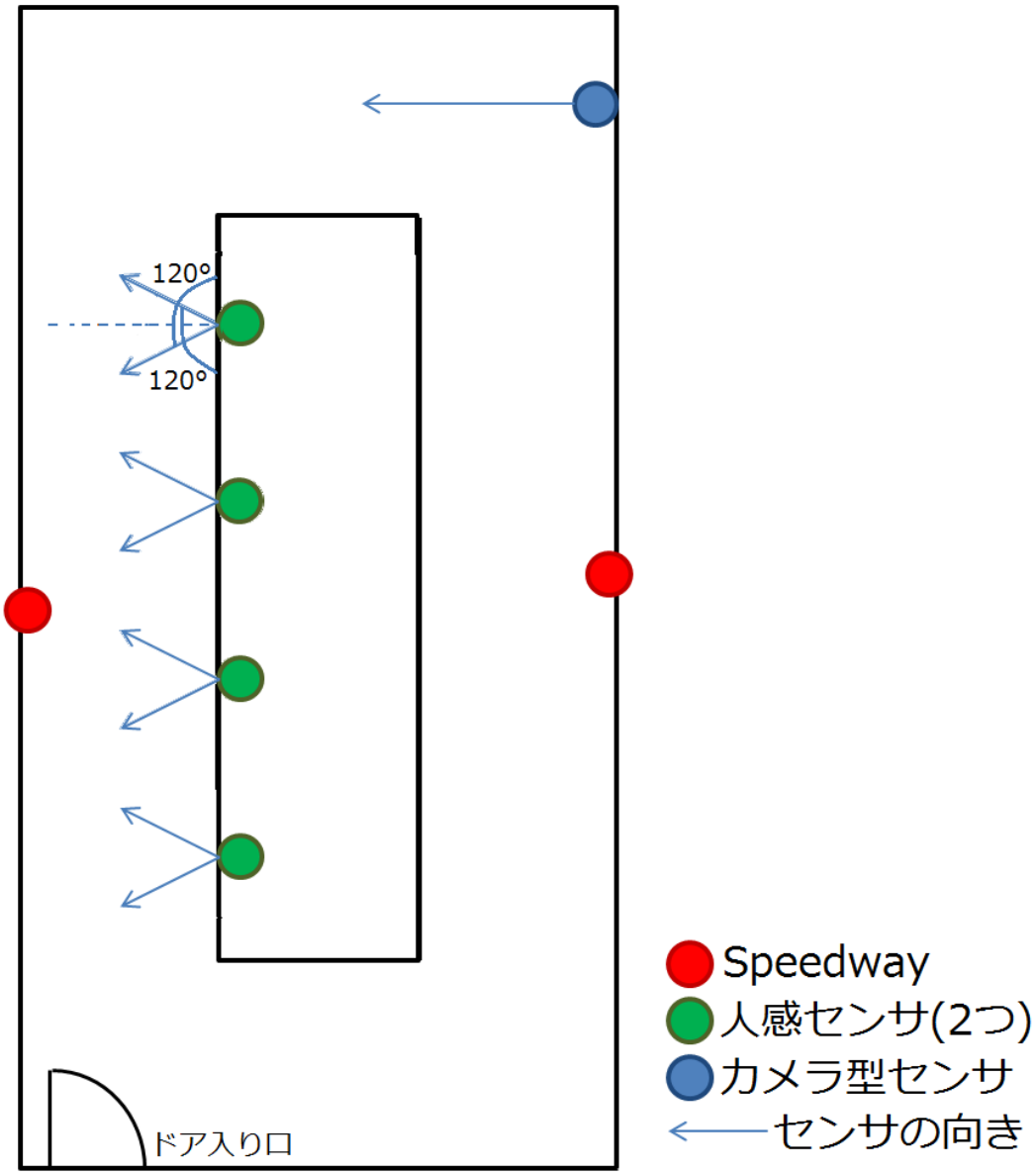


図 6.1: 4F 輪講室及び LS の配置図

コード 6.1: 図 6.1 での設定ファイル (Roomconfig.json)

```
{
    "Room":[
        {
            "name":"4F-SeminarRoom",
            "Room_id":1,
            "floor":4,
            "width":576,
            "depth":1026,
            "height":270,
            "LS":[

                {
                    "name":"Speedway",
                    "LS_id":1,
                    "coordinate":[
                        {
                                "x":1,
                                "y":480,
                                "z":100
                        },

                        {
                                "x":575,
                                "y":550,
                                "z":80
                        }
                    ]
                },

                {
                    "name":"HumanSensor",
                    "LS_id":2,
                    "coordinate":[
                        {
                                "x":370,
                                "y":270,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":330,
                                "y_angle":90
                        },

                        {
                                "x":370,
                                "y":270,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":30,
                                "y_angle":90
```

```
},

{
        "x":370,
        "y":420,
        "z":75,
        "height":700,
        "top_angle":60,
        "x_angle":330,
        "y_angle":90
},

{
        "x":370,
        "y":420,
        "z":75,
        "height":700,
        "top_angle":60,
        "x_angle":30,
        "y_angle":90
},

{
        "x":370,
        "y":570,
        "z":75,
        "height":700,
        "top_angle":60,
        "x_angle":30,
        "y_angle":90
},

{
        "x":370,
        "y":570,
        "z":75,
        "height":700,
        "top_angle":60,
        "x_angle":330,
        "y_angle":90
},

{
        "x":370,
        "y":720,
        "z":75,
        "height":700,
        "top_angle":60,
        "x_angle":330,
        "y_angle":90
},

{
```

```
                                "x":370,
                                "y":720,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":30,
                                "y_angle":90
                            }

                        ]

                    },

                    {
                        "name":"Camera",
                        "LS_id":3,
                        "coordinate":[
                            {
                                "x":575,
                                "y":870,
                                "z":75,
                                "height":700,
                                "top_angle":60,
                                "x_angle":0,
                                "y_angle":90
                            }
                        ]
                    }
                ]
            }
        ]
}
```


ILS を起動させる前に各 LS を起動させてデータを取れる状態にしておき、ILS を起動したら LS からのデータ取得を始める。今回の実験では、ILS は、各 LS から API_LS を介して情報を取得し、LocationData の統合を行い、結果を API_SERVICE を介して出力する、という一連の動作を、1 秒毎に行う設定となっている。今回の実験では、以下のようなパターン分けで行った。

- 実験１： 1 人が 1 つの RFID タグを持ち、Speedway のみ稼働させ、センサ類は稼働させない上で、図 6.2 の A の経路で歩く
- 実験２： 1 人が 1 つの RFID タグを持ち、全ての LS を稼働させた上で、図 6.2 の A の経路で歩く
- 実験３： 2 人のユーザがそれぞれ 1 つの RFID タグを持ち、それぞれ図 6.3 の B,C の経路で歩く

実験１、２においては、同じ経路での条件下で使用する LS の数を変えることで、出力デー

タの差異を測り、比較することを目的とする。実験３においては、ユーザ推論機能についての動作を見ることを目的とする。

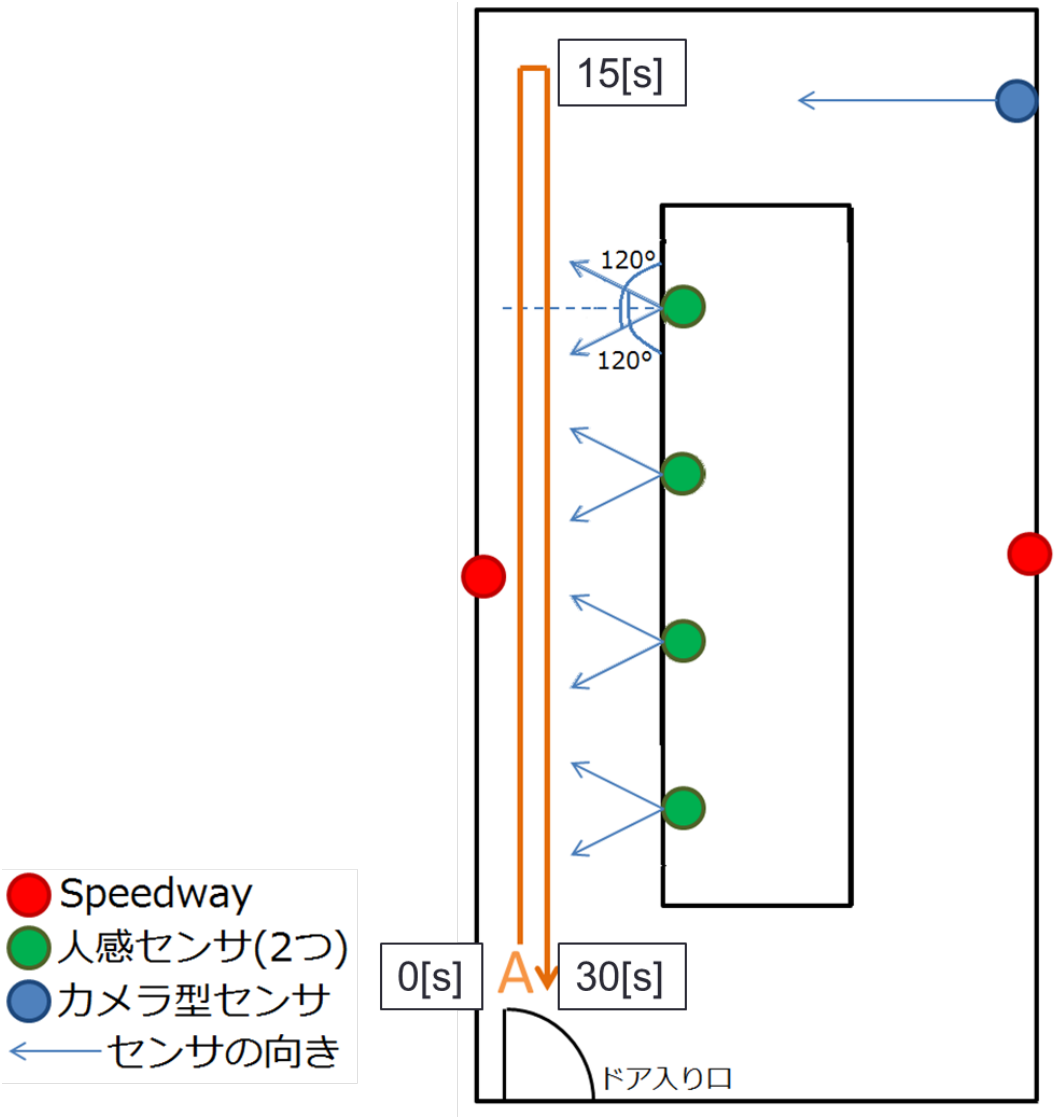


図 6.2: 実験１、２を行う上でのユーザの経路図

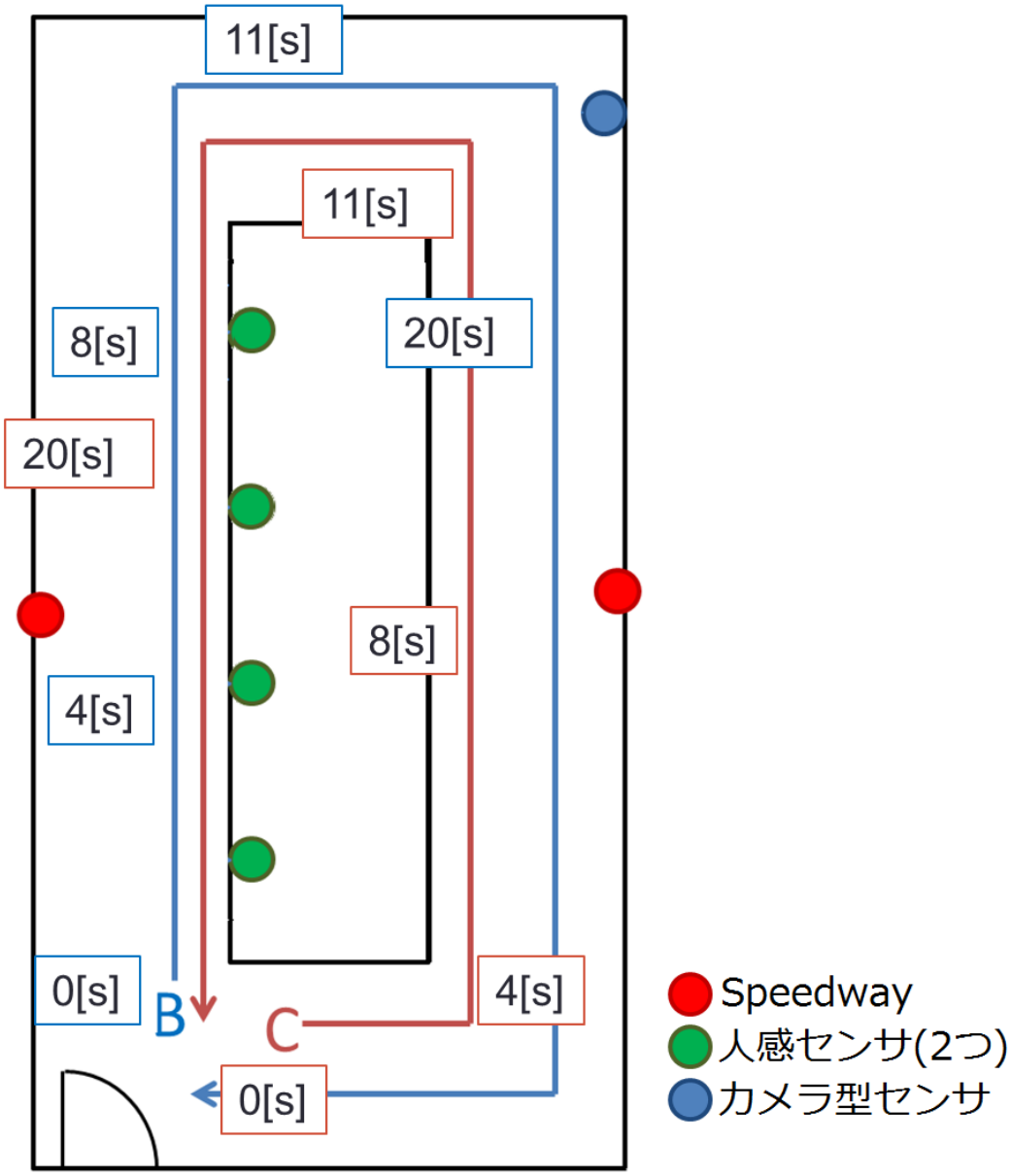


図 6.3: 実験 3 を行う上でのユーザの経路図

6.2　実験結果

各実験の結果を記載する。

6.2.1　実験１・２

実験１、２では実験場所となった 4F 輪講室を図 6.2 の A の経路で歩き、設置した各 LS にデータを取らせた。実験の様子を図 6.4 に示す。

図 6.4: 実験１、２　実験の様子

実験１、２では、ILS を 30 秒間動作させ、図 6.2 において経路 A の折り返しの地点で 15 秒、経路 A の終了地点で 30 秒となるように歩行を行った。

実験１、２において、ILS が 1[s] 毎に出力したデータの精度レベル (1〜6) を示す図を図 6.5 に示す。図 6.5 は実験１、２において、1[s] 毎に ILS が各 LS から取得したデータの統合を行い、API_SERVICE により出力されたデータの精度レベルを表している。

また、実験１、２において、設置した各 LS がユーザ検知を行ったか否かを示す図をそれぞれ図 6.6,6.7 に示す。図 6.6,6.7 において、縦軸は設置した LS それぞれを、横軸は時間 [s] を示しており、グラフ中の●は true、×は false を表している。

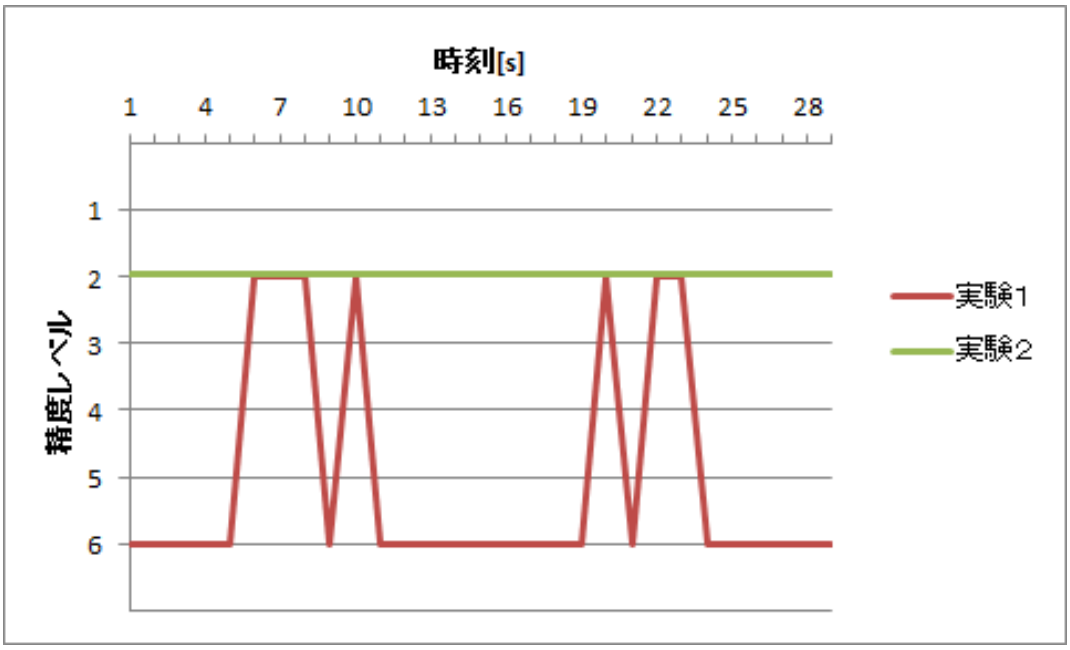
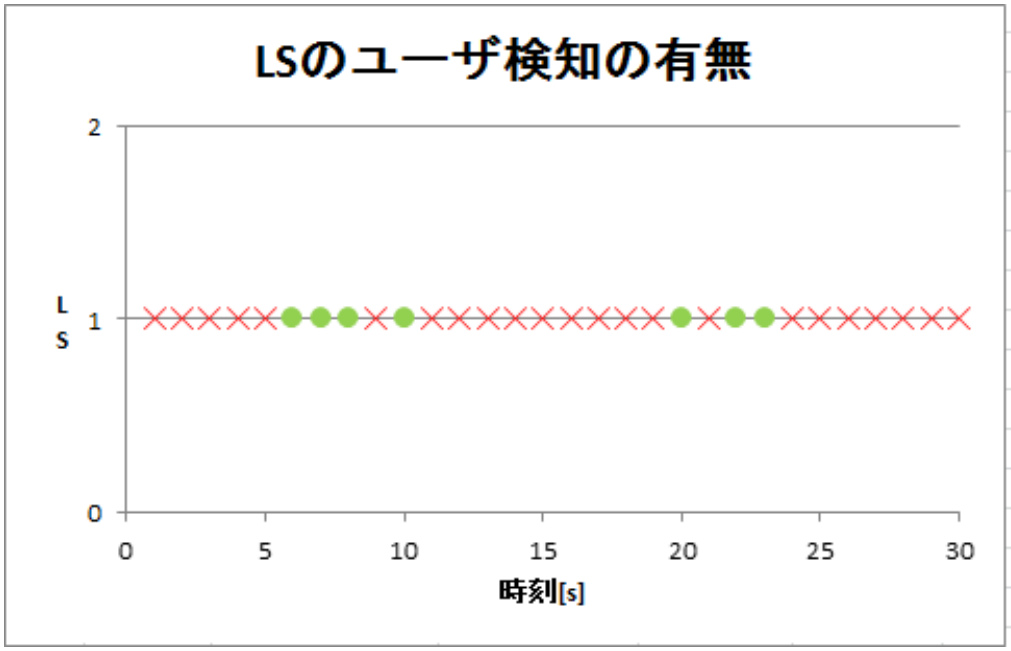


図 6.5: 実験１、２　出力データ精度レベル



※LS(縦軸)

1:Speedway

図 6.6: 実験１　LS のユーザ検知の様子 (Speedway のみ)

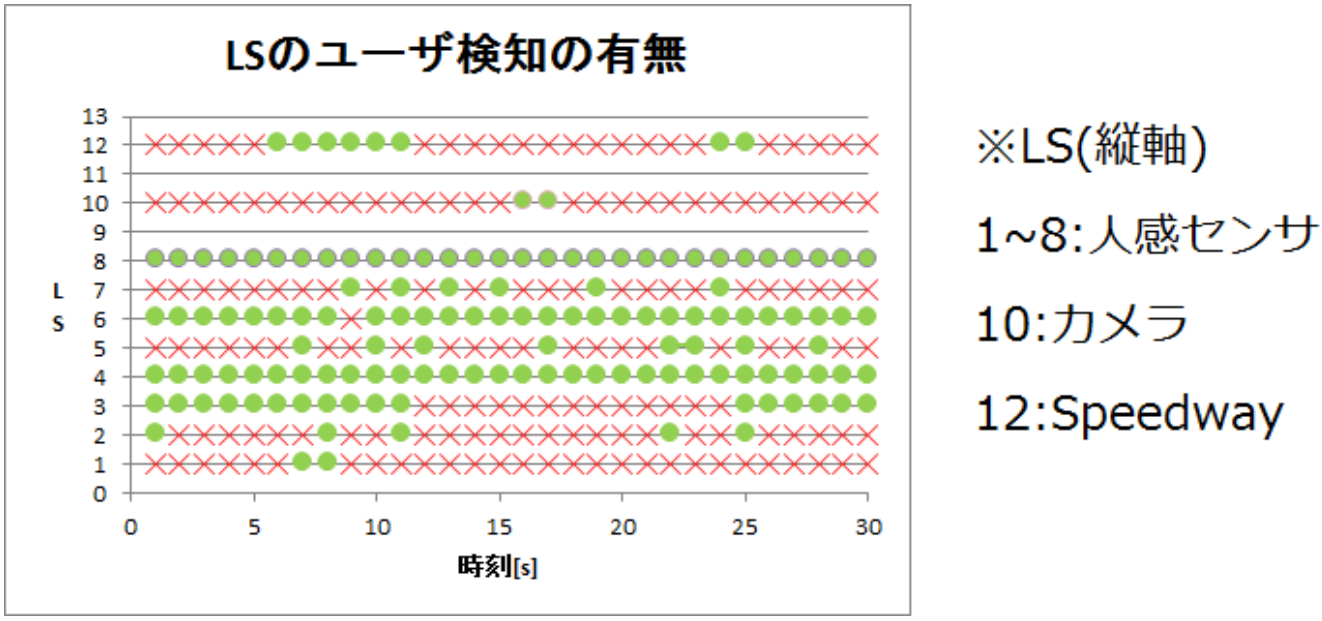


図 6.7: 実験２　LS のユーザ検知の様子

6.2.2　実験３

続いて実験３における出力結果を示す。

実験３では２人のユーザがそれぞれ図 6.3 の B,C の経路で歩き、設置した各 LS にデータを取らせ、ユーザ推論の過程を見た。実験の様子を図 6.8 に示す。

まず、経路 B を歩いているユーザが、Speedway やセンサ類により位置検出される。その時の様子を図 6.9 に示す。図 6.9 〜 6.12 のいずれにおいて、左上の写真はその時の実験の風景、下部の出力はその時の ILS の出力、右上の図は MeshLab で Convexhull を可視化した図であり、またこの図において白線の直方体は実験場所となった輪講室を、左下に位置している桃色の長方形は入り口のドアを示している。

図 6.9 では、実験３開始直後に、経路 B を歩くユーザが Speedway、センサ類により識別、検出され、ILS によって統合され出力されたデータを示している。図 6.9 において、左上の写真は計測時の様子、右上の図はその時の LS の出力を ILS が統合し出力したデータ (Convexhull) を MeshLab で可視化した物であり、下部の文は ILS の出力である。ILS は経路 B を歩くユーザを最初に識別したユーザとして、ユーザ ID を 0 としたユーザ (以後、ユーザ 0 と呼ぶ) とおき、ユーザ 0 の Convexhull を赤色で示している。今後ユーザ 0 の Convexhull を赤色で表す。

図 6.10 は図 6.9 から実験を続けた後の図で、経路 C を歩くユーザが Speedway、センサ類により初めて識別、検出された時の様子を示している。図 6.10 の右上の図 (MeshLab) には新たなユーザを識別・検出したものとして、それまでに識別できているユーザの Convexhull とは違う色で Convexhull を表現している。図 6.10 下部にある ILS の出力にはそれまでに

図 6.8: 実験３　実験の様子

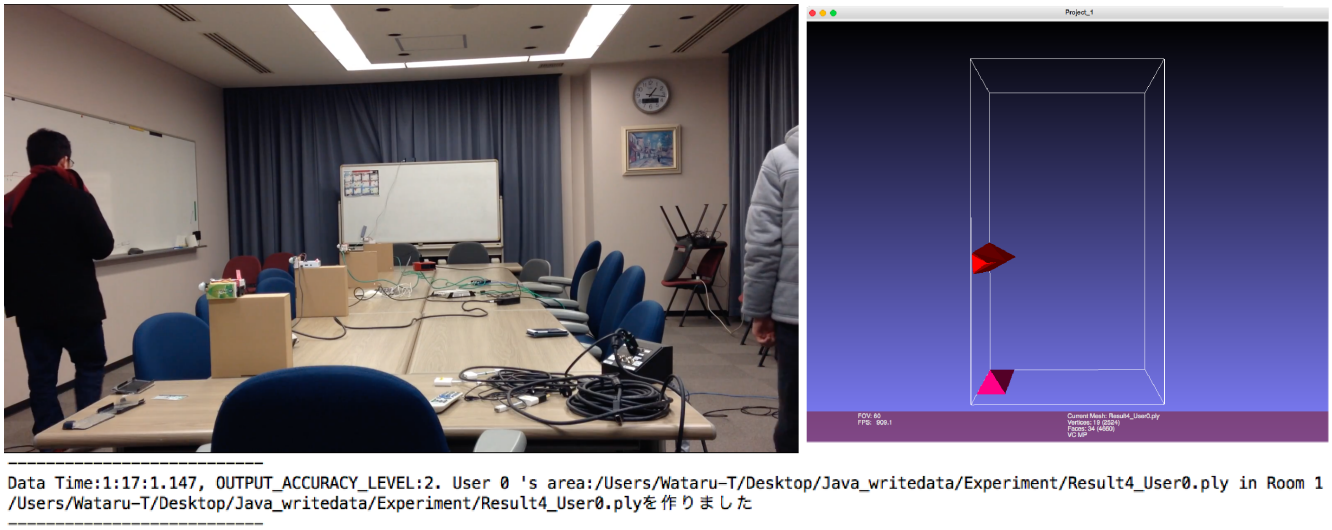


図 6.9: 実験３　位置検出の様子１

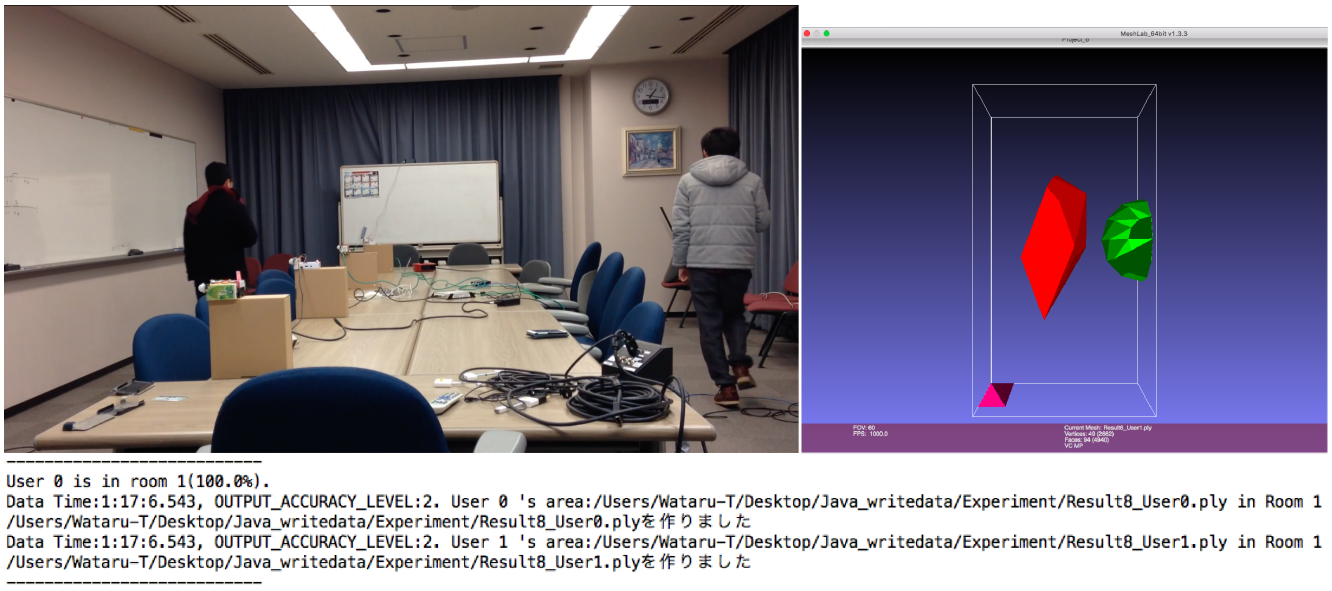


図 6.10: 実験 3　位置検出の様子 2

識別できているユーザそれぞれの出力が得られるが、新しいユーザを識別したことにより、図 6.9 の時よりも出力にユーザが 1 人増えていることが確認できる。ILS は経路 C を歩くユーザを 2 番目に識別したユーザとして、ユーザ ID を 1 としたユーザ (以後、ユーザ 1 と呼ぶ) とおき、ユーザ 1 の Convexhull を緑色で示している。今後このユーザの Convexhull を緑色で表す。ここで、ユーザ 0 はセンサ類の LS にのみ検出されており、不明ユーザの Convexhull が生成されるが、この時点で ILS が識別しているユーザが 2 人で、そのうち 1 人 (ユーザ 1) が特定されている Convexhull を得ていることから、不明ユーザの Convexhull はユーザ 0 であると特定できる。よってこの推論機能により、不明ユーザの Convexhull がユーザ 0 である確率が 100 %であるという情報を付加している様子が図 6.10 下部の ILS の出力から確認できる。図 6.10 右上の Meshlab の図には、ユーザ 0 の位置として赤色の Convexhull が、ユーザ 1 の Convexhull として緑色の Convexhull が表示されている。

図 6.11 は図 6.10 から実験を続けた後の図で、2 人のユーザが交錯している時の様子を示したものである。図 6.11 の右上の図 (MeshLab) はユーザの識別が行えない LS のみの情報で生成した Convexhull なので不明ユーザ（灰色で表している）となるが、ここでユーザ推論の機能が働く。ユーザの位置が特定できない場合には、不明ユーザの Convexhull に対しそれがどのユーザであるかという確率的な指標を付加する。図 6.11 下部にある ILS の出力にはそれまでに識別できているユーザそれぞれに対して出力が得られ、この実験 3 の場合ユーザ 0 とユーザ 1 であるが、他にそれぞれのユーザを特定している Convexhull は無いため、この不明ユーザの Convexhull に対して ILS が推論を行い、この不明ユーザの Convexhull がユーザ 0、ユーザ 1 である確率が 50 %であるという情報を付加している様子が図 6.11 下部の ILS の出力から確認できる。

```
----------------------------
User 0 may be in room 1(50.0%).
Data Time:1:17:10.654, OUTPUT_ACCURACY_LEVEL:3. User -1 is in Room 1
User 1 may be in room 1(50.0%).
Data Time:1:17:10.654, OUTPUT_ACCURACY_LEVEL:3. User -1 is in Room 1
----------------------------
```

図 6.11: 実験 3　位置検出の様子 3

```
----------------------------
User 0 is in room 1(100.0%).
Data Time:1:17:23.4, OUTPUT_ACCURACY_LEVEL:2. User 0 's area:/Users/Wataru-T/Desktop/Java_writedata/Experiment/Result20_User0.ply in Room 1
/Users/Wataru-T/Desktop/Java_writedata/Experiment/Result20_User0.plyを作りました
Data Time:1:17:23.4, OUTPUT_ACCURACY_LEVEL:2. User 1 's area:/Users/Wataru-T/Desktop/Java_writedata/Experiment/Result20_User1.ply in Room 1
/Users/Wataru-T/Desktop/Java_writedata/Experiment/Result20_User1.plyを作りました
----------------------------
```

図 6.12: 実験 3　位置検出の様子 4

図 6.12 は図 6.11 から実験を続けた後の図で、2 人のユーザが交錯を続けた後にそれぞれのユーザが経路を進み続け、その途中でユーザ 1(経路 C を歩くユーザ) が Speedway により識別された様子を示している。もう片方のユーザ 0(経路 B を歩くユーザ) は、他の LS により検知されている。この時、ユーザ 1 は Speedway・センサ類により識別・検知されてユーザが特定された Convexhull を得るが、ユーザ 0 は識別ができない LS からのデータなので不明ユーザのデータとなる。しかし、ユーザ推論の機能によりそれまでに ILS が識別しているユーザが 2 人なので、そのうち 1 人 (ユーザ 1) が特定されている Convexhull を得ていることから、もう一方の不明ユーザのデータはユーザ 0 であると特定できる。よってこの推論機能により、不明ユーザの Convexhull が経路 B を歩くユーザであると見なし、１００％であるという情報を付加している様子が図 6.12 下部の ILS の出力から確認できる。また、図 6.12 右上の図 (MeshLab) から、ユーザ 0 であると断定された不明ユーザの Convexhull が、灰色でなく赤色 (ユーザ 0 の色) になっていることも確認できる。

6.3 評価・考察

実験１、２においては用いるLSの種類を変えることで、得られるデータの比較を目的に実験を行った。

まず実験１において、図6.5から、実験１はデータの出力はILSが動作を開始してから7[s]後からとなり、データの精度レベルは2(ユーザの大まかな区域を示すConvexhullが出力されている)と6(ユーザの位置情報が不明)とで行き来している結果となっている。また、図6.6は実験１において使用したLSがユーザ検知をしたか否かを時刻ごとに示したグラフであり、ILSの動作開始時点を0[s]とし、1[s]の時点から1[s]毎に取得したデータを示したものである。実験１はLSに１つのアンテナを用いたSpeedwayのみを用いているので、図6.6の縦軸(LS)には１つのみ記載となっている。図6.5,6.6から、0[s]〜5[s]の間においてはLS(Speedway)がデータを取得せず、それによりILSもデータを出力していない事が解る。6[s]〜10[s]の辺りにおいては、ユーザがLSに識別・検知され、ILSがここで初めてデータを出力している事が確認できる。またユーザを識別したことにより、ILSはその後のユーザ推論に利用するためユーザを記憶する。11[s]〜19[s]においては、ユーザがLSに検知されてない事が図6.6より解るが、この時点ではILSがユーザを認識しているため、ILSはこのユーザに対して位置情報が不明という精度レベルが6の情報を出力している事が図6.5から解る。20[s]〜23[s]においては6[s]〜10[s]の時と同様にユーザがLSに識別・検知され、ILSがこの間にユーザがいうる大まかな区域(Convexhull)を含んだ精度レベル2の情報を出していることが図6.6から確認できる。24[s]〜においては、LSがユーザを検知できず、ユーザの位置情報が不明という精度レベルが6の情報を返している事が図6.5から解る。

次に実験２において、図6.5から、実験２においてはILSが動作してすぐ精度レベルは2のデータを出力している。図6.7は実験２において使用したLSがユーザ検知をしたか否かを時刻ごとに示したグラフであり、実験１と同様にILSの動作開始時点を0[s]とし、1[s]の時点から1[s]ごとに取得したデータを示したものである。ただし、実験２は実験１よりも多くのLSを設置しているので記載してあるLSも多くなっている。図6.7の縦軸は設置したLSそれぞれを示しており、１８は人感センサ、10はカメラ型センサ、12はSpeedwayを示している。人感センサについては、図6.1において輪講室のドアに近い順から1,2,..8としている。図6.7から1[s]〜5[s]においては、人感センサによりユーザを検知している。この時点ではユーザ識別はできず、ILSもユーザを認識してないが、センサ類のLSが検知した情報により不明ユーザのConvexhullが作成できる。そのため、ILSは不明ユーザがいうる大まかな区域(Convexhull)を含んだ精度レベル2の情報を出力する。6[s]〜11[s]、また24[s]〜25[s]においてはSpeedwayが識別、検知を行い、ILSはユーザ識別情報を含んだ情報を出力できる。この時に、ILSはその後のユーザ推論に使用するユーザ情報を記憶する。ILSの出力としては、ユーザがいうる大まかな区域(Convexhull)を含んだ精度レベル2の情報となる。12[s]〜23[s]、26[s]〜ではセンサ類のLSのみでユーザの位置情報を製作する。経路Aの折り返しの辺りである15[s]の辺りで輪講室奥側をカバーするカメラ型センサが検知している様子が図6.7から解る。この間の情報はユーザ識別ができないLS

のみからの情報なので、普通はユーザの識別ができないが、ユーザ推論機能によりユーザ識別ができない LS からでもユーザ識別が行える事が解る。

この実験 1、２により、LS を単独で用いるよりも複数用いて統合させる方がより精度の高い情報を得られる事が解った。

実験３では 2 人のユーザが輪講室の周りを互いに逆方向に歩き、途中で交錯を行わせる事でユーザ推論の機能の動作確認を行った。ユーザ推論の機能については実験３の過程から推論が行われていることが確認でき、ユーザの識別が行えない LS のみからのデータでも、ユーザ推論の機能を用いることによって確率的な指標を付加することによるユーザ識別を行うことが可能となっている事が確認できた。

しかし、主に実験 2、３において、この実験で得られた Convexhull と実際のユーザの位置とが必ずしも合っているとは限らない結果も得られた。これは統合処理に関する面があると考えているが、他に人感センサの影響によるものもあると考えている。図 6.7 により Speedway 単独よりもセンサ類とを組み合わせることによってより長い時間精度の高いデータを得られることが出力データの精度レベルにより明らかとなったが、同様に人感センサが検知範囲内にユーザがいるいないに関わらず、ずっと true を返し続けていたりなど人感センサの異常とも観れる結果も得られた。

また、今回は 1[s] ごとにデータを出す形式にしたが、LS の時間空間精度差により得られるデータにも差異があると考えるため、より正確なデータを導出するにはこのパラメータについても考慮すべきであると考える。

またアプリケーションなどへの有用性について、今回の実験では API_SERVICE で規定した出力精度レベル (表 3.1) は良いもので２（実験２の結果）であり、Convexhull を作成することでユーザが屋内で大まかにどの辺りにいるかという情報を知ることまでが可能であることが解った。そのため、最高でもユーザの大まかな位置情報以下の精度（出力精度レベル２〜６）を必要とするアプリケーションに対しては対応可能であると考える。また、これよりも精度が高い出力の情報として、今回は検出したユーザの屋内での正確な座標を出力する出力精度レベル 1 を規定したため、今後は出力精度レベル 1 の情報に対応できるように ILS の改良、または LS の設定、導入や使用する LS の増加を行うことが求められる。

第7章　まとめ

本研究ではLSを複数用意し、それぞれにおいて検出したユーザの位置情報を収集、統合してより正確な一つのユーザの位置情報を導出できるような位置情報統合システム(ILS)、かつそれにより得たユーザの位置情報を家庭内向けサービスプラットフォーム及びアプリケーション等に提供できるような技術の提案、実装を行った。具体的には、複数の屋内用の位置検出手法を屋内に設置し、設置した複数の位置検出手法が位置検出を行い、得た結果を位置情報統合システムに送る。位置情報統合システムは各位置検出手法から送信された位置検出結果を受け取り、それらのデータに対し統合処理を行い、一つの位置情報を導出する。その後位置情報統合システムは導出した位置情報を出力する。

また本研究ではユーザの位置を３次元空間上で表現する事とし、屋内の各部屋ごとに独立した座標系を持つものとして扱った。３次元空間内（屋内）において、ユーザがいうる位置をConvexhullという概念を用い表現し、そのために３次元空間上に点を複数与えた時にそれらに対し適切なConvexhullを作成するjavaプログラムの実装、また統合処理をこれらConvexhullの和集合、差集合を取る形でConvexhullの限定を行わせる形で表現し、２つのConvexhullに対して和集合、差集合のConvexhullを作成するjavaプログラムの実装を行った。またLSとして今回はSpeedway(RFID,RSSI), 人感センサ、カメラ型センサを用い、ILSと連携させるための実装をRaspberry Pi等を用い行った。作成したILSにおいて実験を行い、使用するLSの種類をそれぞれ変えた上で出力の比較を目的として行った実験、また実装したユーザ推論機能が正しく動作するかを目的とした実験を行い、評価と考察を行った。

第8章　今後の課題

今後の課題としては、まずILSの設計に関して、今回はLSはILSに対してユーザ識別情報、ユーザとの推定距離や検知の有無といった情報を送ることでLSの検知範囲を示すConvexhullを表現していたが、今後API_LSに対応できないLSが登場し、その導入を検討するときは、それに対応できるようにLSからAPI_LSに送るデータ形式を変えたり、必要に応じて新たな形状のConvexhullの製作、及びそれに応じた変更をILSに行う必要があると考える。ILSに導入が可能なLSとしては、例えば単体の機器を用いてユーザを検知するもの、また場を全体の情報として捉えてその中でユーザを検知するものなどがある。前者においては例えば今回使用したRFIDやセンサ類などが挙げられる。後者においては例えば、今回使用したようなカメラ型のセンサや、他にもサーモグラフィーを用い、観測した熱量からユーザを検知するなどという方法が挙げられる。また、API_SERVICEの出力を特定のアプリケーションなどに対応させ、実際に家庭内向けサービスプラットフォーム及びアプリケーション（例として室内灯のオン・オフやエアコンなどの自動制御など）と連携して実験を行ってみること、また今回は一つの部屋のみでの実験だったため、複数の部屋を用いて実際にそれぞれの部屋にLSを設置し、複数の部屋を連携させて実験を行うことも今後の課題である。また冒頭でGPSは屋内位置測位では行えないと書いたが、GPSはユーザが家庭内にいるかいないかという情報を得ることができ、LSの一つとしてGPSを導入すれば出力データの多様性が増すと考えている。そのため、GPSもLSの一つとしてILSに導入可能であると考えており、この導入も今後の課題である。

次に実験に関して、今回の実験の結果として、得られたConvexhullが大きすぎたり小さすぎたりした場合も見られた。実験でのLSの配置による問題や各LSの時間空間精度差によると思われるが、これらに対して統合のアルゴリズムの改良などの対応を行うことが今後の課題である。また今回の実験の他に、LSの配置、数を変更するなど、今回とは異なる状況下での実験や、多人数での実験を試みる事も今後の課題である。

謝辞

本研究を行うにあたり，終始ご指導ご鞭撻を賜しました丹 康雄教授に深く感謝致します．

また審査員をお引き受け頂いた本学 篠田 陽一教授，本学 リム 勇仁准教授には，本論文を執筆するにあたり多大なご助言を頂きました．深く感謝致します．

副テーマにおいてご指導ご鞭撻を賜りました本学 面和成准教授、金子峰雄教授に感謝致します．

本論文をまとめるにあたりご協力頂いた丹研究室，リム研究室の諸兄に厚く御礼申し上げます．

本研究において、カメラ型センサを提供して頂いた SMK 株式会社様に厚く御礼申し上げます．

最後に，私の研究に対し理解を示して頂き，支えて頂いた家族に感謝を致します．

参考文献

[1] 立石 直樹, ホームネットワークのマルチユーザ化におけるコンテクストを考慮した動的権限管理に関する研究 , 北陸先端科学技術大学院大学, 2012.

[2] 椎尾一郎,RFID を利用したユーザ位置検出システム, 情報処理学会研究会報告 00-HI-88, pp.45-50, 2000.5.12.

[3] Kang Rui Yong, 室内の位置検出プラットフォームを用いた無線干渉と電力の制御に関する研究, 北陸先端科学技術大学院大学, 2016.

[4] Marios Sioutis,Resource Management for Smart Services in the Home, 北陸先端科学技術大学院大学, 2016.

[5] M.Werner,M.Kessel,C.Marouane, ”Indoor positioning using smartphone camera,”in *2011 International Conference on Indoor Positioning and Indoor Navigation*,September 2011,pp.1-6

[6] M. ドバーグ,O. チョン,M. ファンクリベルド,M. オーバマーズ (2010) 『コンピュータ・ジオメトリ ー 計算幾何学：アルゴリズムと応用』 浅野哲夫訳, 近代科学社

[7] Impinj 社 `http://www.impinj.com`

[8] Octane SDK - Impinj Support Portal `https://support.impinj.com/hc/en-us/articles/202755268-Octane-SDK`

[9] SMK 株式会社 `http://www.smk.co.jp`

[10] Math - Commons Math: The Apache Commons Mathematics Library `http://commons.apache.org/proper/commons-math/`

[11] MeshLab `http://www.meshlab.net`