修士学位論文

論文題目 Androidスマートフォンを用いた

視覚障害者向けスマート電子白杖の開発

ふりがな
氏名 塙（はなわ） 暁成（あきなり）

専攻 工学研究科　機械工学専攻

指導教授 金丸　隆志　准教授

修了年月（西暦） 2013年3月

工学院大学大学院

目次

概要

近年白杖などの視覚障害者用補助機器を電子化する試みに注目が高まっている．白杖に関してはセンサやカメラなどを使用した研究が行われ始めているが, データ処理に PC を用いたりするため実用化に至る物は少ない. 本論文ではスマートフォン（Android OS を搭載）を用いてデータ処理部を小型化し，従来の白杖では検知できない危険範囲の障害物を早期認識する電子白杖システムを提案する．

このシステムは，白杖に設置した超音波センサからの距離情報と加速度センサからの加速度情報をスマートフォンに送信し，データ処理を行う．従来の電子補助機器は取得した情報を伝達するだけのものが多く，視覚障害者を混乱させてしまうことがあったが，本研究ではデータ処理に基づく評価値の導入により視覚障害者に有用な情報のみを提供することを試み，実装を行った．

第1章　序論

1.1　研究背景

日本全国の身体障害者数(在宅)は表 1.1 によると平成 18 年において 3,483,000 人いると推計されるが，そのうち視覚障害者は全体の 9%にあたる 310,000 人に上る．過去との比較を行うと視覚障害者数は表 1.1 に示す通り増加傾向にあるため，視覚障害者の生活を支援する政策などが施行されている [1].

本節では視覚障害者の現状や先行研究を踏まえ，本論文の研究背景を示す.

表 1.1　障害の種類別にみた身体障害者数 [2]

年　次	総　数	視覚障害	聴覚・言語障害	肢体不自由	内部障害	(再掲) 重複障害
推　計　数　（単　位：千　人）						
昭和26年	512	121	100	291	－	－
30年	785	179	130	476	－	－
35年	829	202	141	486	－	44
40年	1,048	234	204	610	－	215
45年	1,314	250	235	763	66	121
55年	1,977	336	317	1,127	197	150
62年	2,413	307	354	1,460	292	156
平成3年	2,722	353	358	1,553	458	121
8年	2,933	305	350	1,657	621	179
13年	3,245	301	346	1,749	849	175
18年	3,483	310	343	1,760	1,070	310
対　前　回　比　（単　位：　%）						
昭和26年	－	－	－	－	－	－
30年	153.3	147.9	130	163.6	－	－
35年	105.6	112.8	108.5	102.1	－	－
40年	126.4	115.8	144.7	125.5	－	488.6
45年	125.4	106.8	115.2	125.1	－	56.3
55年	150.5	134.4	134.9	147.7	298.5	124
62年	122.1	91.4	111.7	129.5	148.2	104
平成3年	112.8	115	101.1	106.4	156.8	77.6
8年	107.8	86.4	97.8	106.7	135.6	147.9
13年	110.6	98.7	98.9	105.6	136.7	97.8
18年	107.3	103	99.1	100.6	126	177.1

1.1.1　視覚障害者とは

視覚障害とは，先天性または後天的に起こる病気や外傷，その他の原因によって生じる，永続的な視機能の低下をきたした状態で，もはや治療を行っても機能の改善が認められないものを言う [3]．私たち人間は外部情報を五感で取得しているが，そのうち約 8 割は視覚を用いている．視覚障害者においては感覚器官の中で最も大量の情報を高速に処理できる視覚に障害を引き起こしているため，多くの受け取るべき情報を入手できず，不足した情報の中での生活を余儀なくされている．

視覚障害の比率は先天性のものよりも後天性のものの方がはるかに多い．その原因となった疾患は表 1.2 に示す通り網脈絡膜・視神経系疾患である [2]が，その代表的なものとして糖尿病性網膜症[1]や緑内障が挙げられる．糖尿病は日本人の 7 人に一人はかかると言われる成人病であり，年々増加傾向にある（表 1.3） [4]．また，緑内障についても 40 歳以上の日本人には，20 人に一人の割合で患者がいるとされている [5]．このように視覚障害とは晴眼者である我々にも身近な障害である．

[1] 糖尿病の合併症として発症．網膜及び硝子体に出血を生じ，進行すると網膜剥離を起こして失明することがある．

表 1.2　障害の種類別にみた身体障害の原因疾患［2］

（単位：千人）

	総　数	視覚障害	聴覚・言語障害	肢体不自由	内部障害	（再掲）重複障害
総　数	3,483 (100.0)	310 (100.0)	343 (100.0)	1,760 (100.0)	1,070 (100.0)	310 (100.0)
脳性まひ	54 (1.6)	4 (1.3)	－ (－)	50 (2.8)	－ (－)	11 (3.5)
脊髄性小児まひ	43 (1.2)	－ (－)	2 (0.6)	42 (2.4)	－ (－)	5 (1.6)
脊髄損傷Ⅰ（対まひ）	33 (1.0)	1 (0.3)	1 (0.3)	31 (1.8)	1 (0.1)	5 (1.6)
脊髄損傷Ⅱ（四肢まひ）	24 (0.7)	－ (－)	1 (0.3)	23 (1.3)	－ (－)	2 (0.6)
進行性筋萎縮性疾患	21 (0.8)	－ (－)	2 (0.6)	20 (1.1)	－ (－)	2 (0.6)
脳血管障害	273 (7.8)	7 (2.3)	11 (3.2)	254 (14.4)	－ (－)	51 (16.5)
脳挫傷	11 (0.3)	2 (0.6)	1 (0.3)	9 (0.5)	－ (－)	2 (0.6)
その他の脳神経疾患	73 (2.1)	6 (1.9)	9 (2.6)	57 (3.2)	1 (0.1)	16 (5.2)
骨関節疾患	238 (6.8)	－ (－)	2 (0.6)	234 (13.3)	2 (0.2)	10 (3.2)
リウマチ性疾患	97 (2.8)	－ (－)	1 (0.3)	94 (5.3)	2 (0.2)	7 (2.3)
中耳性疾患	32 (0.9)	1 (0.3)	27 (7.9)	2 (0.1)	2 (0.2)	1 (0.3)
内耳性疾患	45 (1.3)	－ (－)	43 (12.5)	－ (－)	2 (0.2)	8 (2.6)
角膜疾患	19 (0.5)	19 (6.1)	－ (－)	－ (－)	－ (－)	6 (1.9)
水晶体疾患	11 (0.3)	11 (3.5)	－ (－)	－ (－)	－ (－)	1 (0.3)
網脈絡膜・視神経系疾患	84 (2.4)	82 (26.5)	－ (－)	2 (0.1)	1 (0.1)	7 (2.3)
じん臓疾患	163 (4.7)	2 (0.6)	－ (－)	－ (－)	161 (15.0)	14 (4.5)
心臓疾患	350 (10.0)	1 (0.3)	－ (－)	1 (0.1)	349 (32.6)	11 (3.5)
呼吸器疾患	56 (1.6)	1 (0.3)	3 (0.9)	1 (0.1)	51 (4.8)	6 (1.9)
ぼうこう疾患	20 (0.6)	－ (－)	－ (－)	－ (－)	20 (1.9)	1 (0.3)
大腸疾患	51 (1.5)	－ (－)	－ (－)	1 (0.1)	51 (4.8)	－ (－)
小腸疾患	4 (0.1)	－ (－)	－ (－)	－ (－)	4 (0.4)	－ (－)
後天性免疫不全症候群	2 (0.1)	－ (－)	1 (0.3)	－ (－)	1 (0.1)	－ (－)
その他	286 (8.2)	48 (15.5)	35 (10.2)	181 (10.3)	21 (2.0)	21 (6.8)
不　明	78 (2.2)	14 (4.5)	30 (8.7)	30 (1.7)	3 (0.3)	7 (2.3)
不　詳	1,414 (40.6)	112 (36.1)	175 (51.0)	728 (41.4)	399 (37.3)	118 (38.1)

（　）内は構成比（%）

表 1.3　糖尿病患者の推移 [4]

	平成 9 年	平成 14 年	平成 19 年
「糖尿病が強く疑われる人」	約 690 万人	約 740 万人	約 890 万人
「糖尿病の可能性を否定できない人」	約 680 万人	約 880 万人	約 1,320 万人
「糖尿病が強く疑われる人」と「糖尿病の可能性を否定できない人」の合計	約 1,370 万人	約 1,620 万人	約 2,210 万人

1.1.2　白杖について

図 1.1 に示す白杖の正式名称は盲人安全杖であるが [6], 一般的には白杖と呼ばれることが多い．白杖の機能は大きく分けて 3 つあげられる．

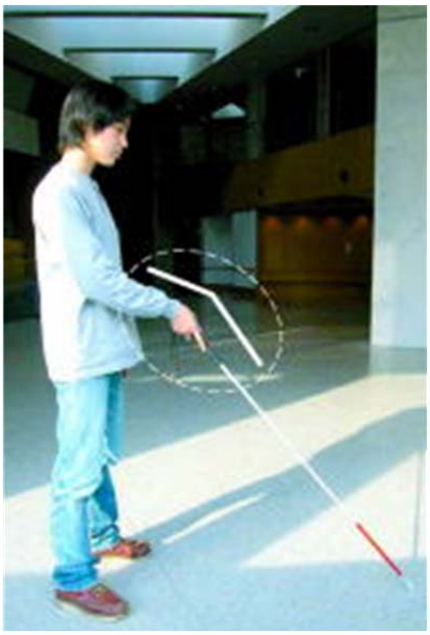
1. 障害物からの防御
2. 歩行面の情報収集
3. 存在の周囲への通知

白杖の使用方法・基本姿勢を図 1.2 に示す．その際，白杖は身体の正面に構え，グリップはへその位置から 20cm ほど前に出す．グリップは逆「く」の字になるように構える．石突きが肩幅から 2.5cm 出る程度振りながら歩行面の情報を収集していく [7].

視覚障害者には白杖の携帯義務が道路交通法第 14 条 [8]で規定されており，ドライバや他の歩行者・警察官への注意喚起の意味合いがある．また，白杖を用いた階段や段差などの危険個所認識はとても重要な役割を果たす．重度の視覚障害者のほとんどは，白杖を用いて前方の状況を確認しているが，その範囲は白杖の届く 1.5m 程度の範囲に限られ，それより遠方の情報を得ることは不可能である．

また，視覚障害者が単独歩行を行う際最も重要なことは，自らの歩く経路を頭の中に歩行地図（メンタル・マップ）として正確に認識できることである．メンタル・マップと白杖を併用し，目的地まで歩行を行う．

図 1.1　白杖



1) 握り方

2) 構え方

3) グリップの位置

4) 振り幅

図 1.2 白杖使用時の基本姿勢 [7]

1.1.3　電子補助器具の先行研究

表 1.4 に示す通り，視覚障害者の週 2～3 回以上の外出の頻度は，他の障害を持つ方に比べて低いことが分かる．生後初期からの視覚障害者はもとより，過去に日常生活になんら問題のなかった中途失明者でも急激な視覚情報不足に陥ることで，見えない世界へ踏み出すのが怖くなり外出を控えることが多い．

そのため，近年白杖などの視覚障害者用補助機器を電子化する試みに注目が高まっている．その要因として視覚障害者の自立による社会的地位向上を目指す動きが挙げられる．上述した社会的，物理的な問題に対して工学的なアプローチから視覚障害者の社会自立を支援するのが電子補助機器の役割である．また，個々人の尊厳を尊重し，自立した生活と社会への積極的参加を実現するために，補助機器の使用は必要不可欠である．以下で視覚障害者用補助機器の先行研究例を紹介する．

表 1.4　障害の種類別にみた外出の状況 [2]

	総　数	外　出　あ　り					外出なし	回答なし
		ほぼ毎日	週2～3回	月2～3回	年に数回	小　計		
総　数	4,263	1,519	1,273	687	412	3,891	235	137
	(100.0)	(35.6)	(29.9)	(16.1)	(9.7)	(91.3)	(5.5)	(3.2)
視覚障害	379	111	113	83	40	347	24	8
	(100.0)	(29.3)	(29.8)	(21.9)	(10.6)	(91.6)	(76.3)	(2.1)
聴覚・言語障害	420	175	115	59	31	380	22	18
	(100.0)	(41.7)	(27.4)	(14.0)	(7.4)	(90.5)	(5.2)	(4.3)
肢体不自由	2,154	679	644	356	256	1,935	150	69
	(100.0)	(31.5)	(29.9)	(16.5)	(11.9)	(89.8)	(7.0)	(3.2)
内部障害	1,310	554	401	189	85	1,229	39	42
	(100.0)	(42.3)	(30.6)	(14.4)	(6.5)	(93.8)	(3.0)	(3.2)

(　) 内は構成比 (%)

・「スマート電子白杖」秋田精工株式会社製作 [9]

秋田精工株式会社と秋田県立大学が共同開発した多機能化白杖が既に製品化されている．これは白杖に距離センサを取り付け，前方の障害物の高さにある突起物を感知し，グリップ及びリストバンドの振動により視覚障害者に伝えるものである．センサ 1 個タイプは頭部前方 2m の障害物を感知し，センサ 2 個タイプは正面と頭部前方 2m の障害物を感知する．

スマート電子白杖の一番のメリットは既存の白杖では認識できなかった頭部前方の障害物認識を可能にしたことに加え，その価格である．通常このような補助機器は製品化されると数十万と高価になってしまい，なかなか障害者の方が使用できない現状があった．しかし，このスマート電子白杖はセンサ 1 個タイプが本体 30,000 円，センサ 2 個タイプが本体 43,000 円と安価に入手することができ，また，秋田県では 2011 年度より「視覚障害者用電子白杖購入助成事業」に基づき，購入者への助成を行っている [9] [10]．しかし，段差がある場合や下り階段，障害物の方向などの認識には不安が残る．

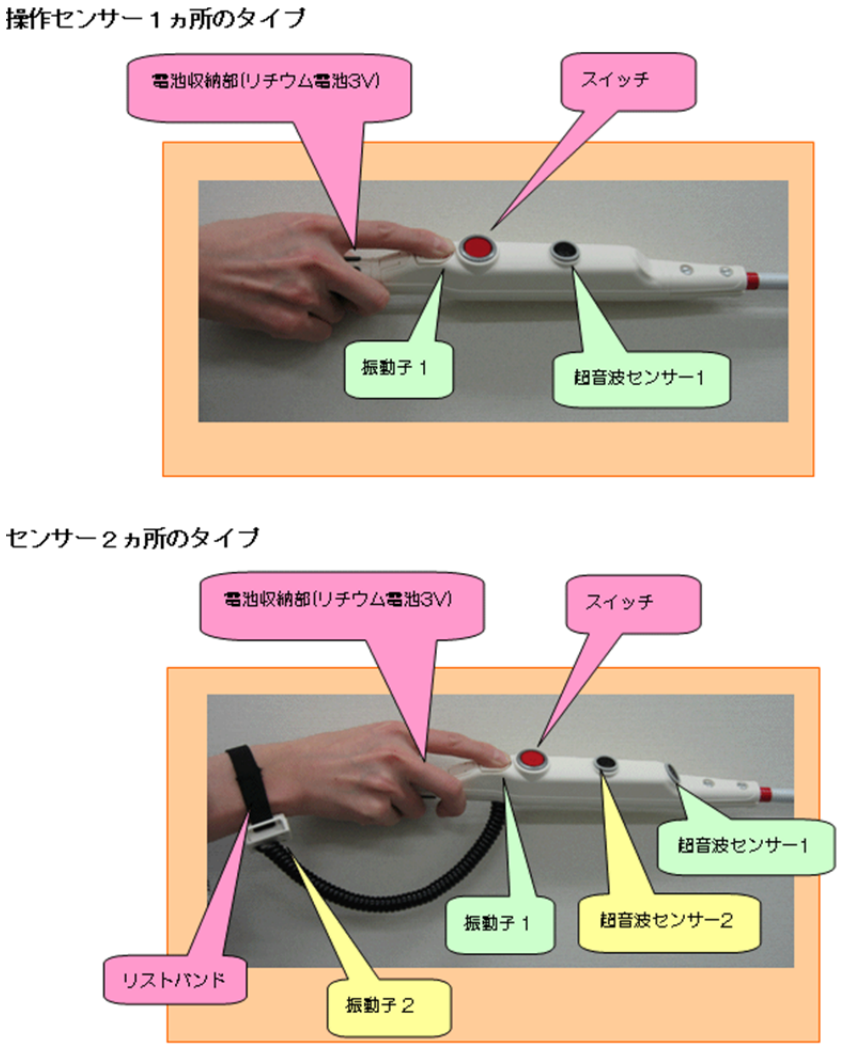


図 1.3　操作部外観 [9]

・「Navigational Aids for the Visually Impaired」University of Konstanz [11]

ドイツのコンスタンツ大学修士学生 2 人が Microsoft のモーションセンサ Kinect を視覚障害者向けの補助装置として利用する試みを行っている．ヘルメットに取り付けた Kinect によって壁や障害物までの距離を測定し，腰に巻きつけた振動装置で着用者に警告するシステムである．また，壁やドアに貼られたシンボルマークを検出し，距離などを音声で伝えることも可能である．

装置の大きさやデータ処理部にノート PC を使用しているなど実用化にはまだまだ課題はあるが，既存技術の応用から生まれた新たな視覚障害者用補助機器の一例だといえる．

図 1.4　ラップトップ専用バック［11］

図 1.5　Kinect の装着方法［11］

・「Ariadne GPS」ジョヴァンニ・チャッフォーニ氏［12］

「Ariadne GPS」は視覚障害者が地図を調べ，自分の現在地を知るのを補助するアプリケーション（以下アプリ）である．「VoiceOver」によって，視覚障害者が音声フィードバックを用いて地図を調べることを可能にし，スマートフォン内蔵の GPS を用いて，行きたい場所の近くに来たときにユーザー通知を行うことができる．また，地図を触るだけでその場所の住所や通りの名前，川の名前などを音声で伝えてくれる．スマートフォンの操作は触ったボタンをすべて音声で通知してくれ，視覚障害者の方でも迷わず使用できる．地図による視覚障害者の誘導や現在地の提供機能，お気に入りの場所を登録し近づくと通知をしてくれる機能などがある．また，バスなどに乗っている時に目的地から数百 m の距離まで来ると音声で教えてくれる．視覚障害者支援のアプリは様々なものが開発されているが，地図を探索する機能があるのはこのアプリだけである．このアプリはジョヴァンニ・チャッフォーニ氏が一人で手掛けたものであるが，そのアイデアや技術力にアップルのティム・クック CEO が WWDC[2]に彼を招待したほど革新的なものである［12].

視覚からの情報がない中で，単独歩行を行うのがどれほど大変で危険なものなのかは晴眼者の我々には想像することはできない．だからこそ，視覚障害者にとって正確な自己位置の認識（メンタルマップ）は安全な単独歩行を行う上で重要なことである．しかし，私たちが地図を眺めることにより得ている情報を視覚障害者の方々は全く利用できないため，メンタルマップを構築することはとても大変なことである．

その課題を解決するために開発されたのがこのアプリである．スマートフォン内部には GPS などの各種センサがあり，また，インターネットへの接続により多種多様な情報を瞬時に手に入れることができる．スマートフォンという高性能の携帯電話は，まさにハンデを持つ人にとってより有益であり，新しいテクノロジーは視覚障害者の社会参加や自立の根本的な手段となることが可能である．

図 1.6 「Ariadne GPS」アプリケーション［12］

[2] Apple 主催のデベロッパー向けのイベント.

1.1.4　視覚障害者からの聞き取り調査

福祉機器や福祉器具の開発において，研究レベルのものは多く存在するが，製品として世の中に出ない，製品化しても普及するものが少ないという現状がある．要因としては，開発者側と当事者の方との間でうまくコミュニケーションが取られずに使えないものや使いにくいものが製作されてしまうということがある [13]．視覚障害への理解を深め，現在の補助機器の問題点を明確にするために聞き取り調査を行った．

まず聞き取りをお願いしたのは，茨城県日立市在住の 60 歳代の男性で生まれつき全盲の方である．平成 23 年 9 月に行った聞き取りの内容は以下の通りである．

・一人暮らしのため，週に 3 回介助ヘルパーに身の回りのことを手伝いに来ていただいている．
・外出に関してはガイドヘルパー[3]の支援時間は限られているため，必要最低限の外出の際にしか使用できない．そのため，大抵の外出は単独で行わなくてはならない．
・ニュースなどの情報を得るのはラジオである．携帯電話は使用するが PC は使用しない．
・夜の外出は周りに気づいて貰えないなど危険なためできない．
・白杖の検知範囲以外の障害物は認識できないため，何度もぶつかる．
・視覚障害者用電子補助機器の使用に関してはぜひ使ってみたい．しかし，どのような製品があるかが分からない．また，価格に関しても高額のため使用できない．
・屋外での歩行の際は車の音を頼りにしているが，信号や交差点などは分からない．
・日常生活の中では自分の感覚を頼って生活している．

次に平成 24 年 11 月に茨城県立視覚障害者福祉センター[4]で歩行訓練士[5]の方にお話を伺い，視覚障害者の方の生活実態や興味関心などに関しても具体的に教えて頂いた．内容は以下の通りである．

・視覚障害者が歩行を行う際，自己位置の認識（メンタルマップ）をいかに正確にするかが重要．メンタルマップを育成するための歩行訓練を重視している．
・既存の電子補助機器については便利な物もある．特に秋田精工製のスマート電子白杖は頭上の障害物が分かり便利である．
・細かすぎる（不要な）情報は混乱をするだけなので出さないで欲しい．
・下り坂や階段の入り口などが早期に分かると助かる．
・視覚障害者の方は健常者とできるだけ同じ生活を行いたいと考えている．そのため，電子機器の使用に関してとても興味を持っている．特にスマートフォンなどは視覚障害者センターなどでも体験会や講座が何度も開かれるほど積極的である．

---

3 視覚障害者を目的地に送迎する人．現在では「障害者自立支援法」の制度にも位置付けられている．

4 身体障害者福祉法第 34 条に基づいて，視覚障害者の厚生を援護し福祉の向上をはかるため，社会福祉法人茨城県視覚障害者協会が茨城県から管理・運営を委託された施設である．

5 視覚障害生活訓練等指導者の通称．目の見えない人が白杖を持って不安なく歩けるようにする訓練を専門に行う人．

以上の聞き取り調査より，視覚障害者は白杖の検知範囲以外の障害物は認識できないため，障害物を早期に認識できる電子補助機器の使用に積極的であるが，有益ではない細かすぎる情報提供は望んでいないことが分かる． 視覚障害者にとってどのような情報が有益かを考慮して開発を行わなければならない．また，視覚障害者はできるだけ健常者と同じ生活環境で過ごしたいと考え，スマートフォンの使用に積極的である．この背景には視覚障害者センターでの講習会や 2.4 で述べる簡単に使用できるスマートフォンなどが製品化されていることが挙げられる．これらの意見を本研究の参考とする．

1.2　本研究の目的

これまで見たように，視覚障害者用電子補助機器の開発が進んでいるが，障害者にとって有益な周辺情報の提供が十分ではなく，まだまだ周囲の人々の援助に頼ることが多いのが現状である．また，視覚障害者にとって不要な情報の提供は混乱を生じさせることが分かったが，これは取得情報（データ）のみによる情報提供では具体性に欠けるからであると考える．

そこで，視覚障害者にとっての必要な情報を決定するために，人間による視覚障害者支援方法を参考にする．周囲から視覚情報不足を補う支援の方法として「道で障害物などを知らせる場合は，視覚障害者自身の向きにしたがって左右を教える」，「距離を指摘する際には，具体的に何メートルかを指定して告げる」が挙げられるため，本研究ではこの二つを視覚障害者にとって有益な情報と定義し，障害物方向及び距離の認識を目指す．

本研究の目的である障害物方向を特定するために，白杖に外部の超音波センサと加速度センサを搭載し，障害物までの距離と白杖の動きを取得する．また，そのデータを自由度の高い無線通信を用いてデータ処理部に送信し，視覚障害者に障害物方向及び距離の有益な情報を提供できるシステム構築を目指す．データ処理部には各種センサを内蔵している多機能性や Android OS の高い拡張性を有しているスマートフォン（Android OS を搭載）を用いることにより， 新たな視覚障害者用電子補助機器の一例を提案する．

1.3　本論文の構成

本論文の構成を以下に示す.

まず，第 1 章では研究背景と本研究の目的について述べる．電子白杖とその他の補助機器の先行研究や視覚障害者の実態についても述べる.

第 2 章では本研究の準備段階として，システムの概要や RT-ADKmini や各種センサについての詳細なスペックについて示す.

第 3 章ではファームウェアについて述べる．データの前処理などについて詳細を示す.

第 4 章では Java アプリケーションについて示す．フローチャートによるシステムの流れや，データ受信から内部メモリの保存までについての詳細を示す.

第 5 章では評価値についての理論を述べる．sin 波への近似方法や評価値理論式に関して示し，障害物方向の判別基準についても述べる.

第 6 章では評価実験について述べる．評価値による障害物方向の特定について示す.

第 7 章では電子白杖のカスタマイズ機能での利用について述べる．ファームウェアと Java アプリケーションにおける変更点，使用者への伝達方法について示す.

第 8 章ではカスタマイズした白杖を用いた評価実験について述べる.

第 9 章では本論文で得られた成果を要約し，総括とする.

第2章　システムの概要

2.1　システム全体構成

図 2.1 にシステムの全体構成を示す．システムには第 2 章～第 6 章では超音波センサ 1 個版（図 2.1 上）を使用し，第 7 章以降では超音波センサ 4 個版を用いる（図 2.1 下）．障害物方向を特定するため，白杖の動きを 3 軸加速度センサで，障害物までの距離を超音波距離センサによって計測する．各センサ値を同期させて計測したデータは，スマートフォンとの通信を可能にする PIC24FJ64GB002（以下 PIC）を搭載したアールティ社製の RT-ADKmini というマイクロコンピュータボード（以下マイコン）を使用して送受信を行う．その際，スマートフォンとマイコンは無線（Bluetooth）で接続している．なお，PIC とスマートフォンとの間で Bluetooth 通信を行うために，PIC のファームウェアとして hrdakinori 氏による BT _DROID を活用した [14].

本研究はこのスマートフォンを用いて電子白杖を小型化し，従来の白杖では検知できない危険な路面状況や障害物方向を認識することを目指している．システムに高い計算能力を組み込む場合, スマートフォンはノート PC に比べ携帯性に優れているというメリットがある．また，スマートフォン自体は白杖から独立しているので，白杖自体の小型化も図れる．次節でそれぞれの仕様を示す.

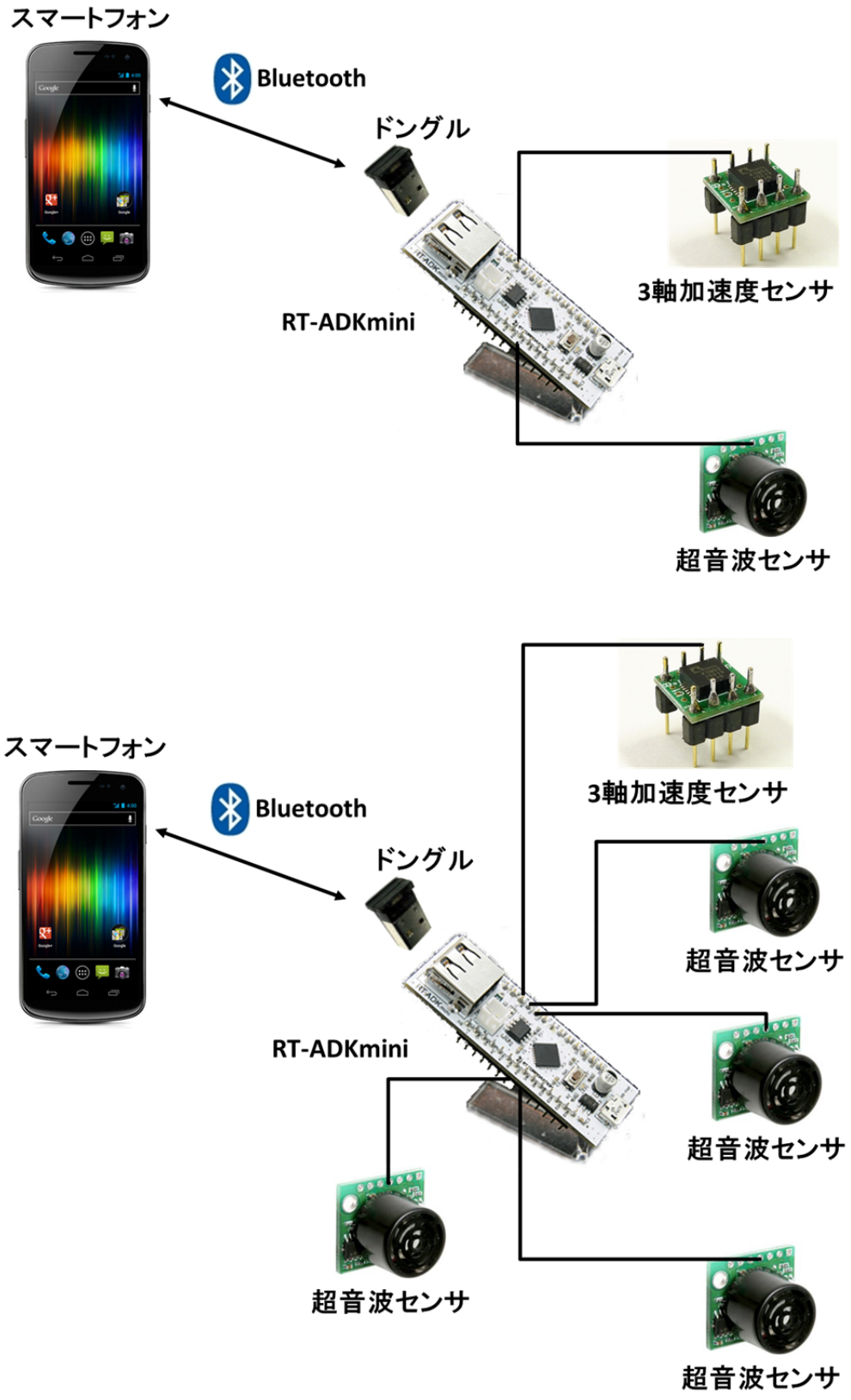


図 2.1　システムの全体構成　上）超音波センサ 1 個版　下）超音波センサ 4 個版

2.2　RT-ADKmini

RT-ADKmini とは PIC24FJ64GB002 を搭載した 28 ピン DIP 形状のマイクロコンピュータボードである．USB のホスト側になるタイプ A の USB コネクタと 5V の電源を得るためのミニ B の USB コネクタが存在する．サイズは 62×21×18mm，重さは 11g である．

本論文では，Bluetooth 通信によりボードとスマートフォンを接続させるため，タイプ A の USB ポートに Buffalo 製の BSHSBD03 ドングルを用いて使用している（図 2.3）．ドングルの仕様を以下に示す（表 2.1）．

表 2.1　Bluetooth ドングルの仕様 [15]

・無線インターフェース	Bluetooth Ver 2.1 +EDR Class2 準拠
・USB インターフェース	USB2.0
・通信出力	最大 2.5mW (class Ⅱ)
・通信距離	約 10m (使用環境により異なる)

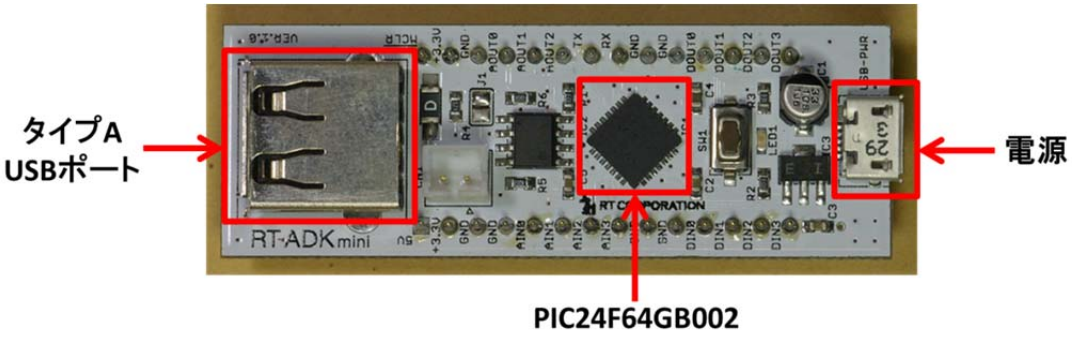


図 2.2　RT-ADKmini 詳細図

図 2.3　Bluetooth ドングルの概観（Buffalo BSHSBD03）

2.3　計測センサ詳細

計測センサには超音波センサと 3 軸加速度センサを使用している．それぞれのセンサの詳細を以下に示す.

2.3.1　超音波センサ

はじめに超音波センサの紹介を行う．超音波とは，周波数（1 秒間の振動数）が 20kHz（1 秒間に 20,000 回振動する）を超える音波のことを呼び，人の耳には聞こえない非可聴音域の音波である．可聴音は世に溢れているが，非可聴音である超音波は自然界ではあまり存在せず, 誤作動を起こしにくいという利点がある．この超音波を一定周期で発生させ，発生時点からこの超音波が対象で反射して返って来るまでの時間を計り，音の速さを用いて距離を計算するのが超音波センサである [15].

対象物までの距離を D（m），その時の音速を C（m/s），超音波の発信から受信までの時間を t（秒）とすると，距離は

$$\mathrm{D} = \mathrm{C} \times \frac{t}{2} \tag{2.1}$$

で求めることができる．ここで空気中の音速は，温度や湿度などの環境条件によって変化し，特に温度変化による影響を受け，温度が高いほど音速が早くなる性質がある．乾燥空気中の温度を T（℃）とした場合，音速 C（m/s）は,

$$\mathrm{C} = 331.45 + 0.607 \times \mathrm{T} \tag{2.2}$$

で求めることができる. 一般的に用いられる空気中の音速は 340 (m/s) だが, これは約 15℃の時の値であり, 仮に 0℃まで減少した場合には約 331(m/s)となり 2.7%の誤差が生じる. このため超音波受信器に温度センサを内蔵して, 温度補正を行うことが一般的である [16]. しかし，超音波センサを白杖に装着して使用する場合には計測器の基準位置自体が不安定となることから，補正をしなくても特に問題ないと判断する.

本研究では自動キャリブレーションや測定距離の安定性が高いことなどから MaxBotix Inc.の LV-MaxSonar-EZ1 を使用し，出力にはアナログ出力を利用する．超音波センサの仕様と概観を以下に示す（図 2.4，表 2.2） [17].

表 2.2　超音波センサの仕様［18］

・測定距離	0～6.54m
・測定精度	1inch（2.54cm）
・サンプリング周波数	20Hz
・超音波周波数	42kHz
・出力：パルス幅	147μs/inch
アナログ電圧	9.8mV/inch
シリアル	9600bps
・電源	2.5V～5.5V

電源電圧には 3.3V を用い，RT-ADKmini ボードから給電している．センサのサンプリング周波数は 20Hz となっており，本システムのサンプリング周波数は 10Hz であるため 2 回に 1 回のデータ取得となる．

図 2.4　超音波センサの概観［17］

2.3.2　加速度センサ

加速度とは，単位時間当たりの速度の変化率であり，加速度センサはこの変化率を電圧の変化として出力する．本論文では加速度センサには 3 軸加速度センサ KXM52-1050 モジュールを使用した（図 2.5）．これはカイオニクス社製チップ型 3 軸加速度センサ KXM1050 を基板に取り付け，使いやすくモジュール化したものである．この加速度センサには 3 軸に対応しており，それぞれの軸に対する加速度に応じた電圧を出力してくれる．センサ仕様を表 2.3 に示す [18].

表 2.3　加速度センサの詳細 [19]

・測定レンジ	±2g
・感度	660mV/g±5%（電源 3.3V 時）
・0g 出力	1.65V±167mV（電源 3.3V 時）
・電源電圧	2.7～5.5V（標準 3.3V）

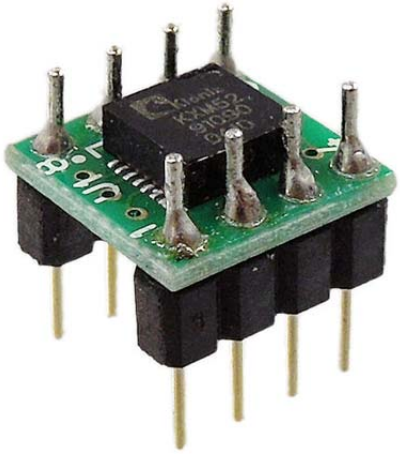


図 2.5　加速度センサの概観 [18]

加速度センサの 3 軸の方向を図 2.6 に示す．この加速度センサを白杖に取り付けることにより白杖の動きを検知する．本研究では加速度センサを用いて白杖の左右の動きを認識することを目指しているため，図 2.7 に示す Y 軸方向の加速度のみを評価値計算[6]に使用する．

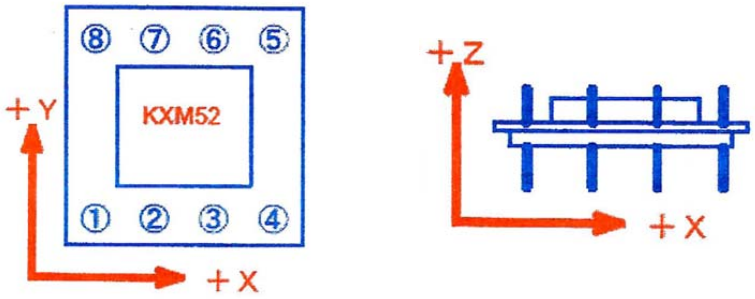


図 2.6　加速度センサの軸方向

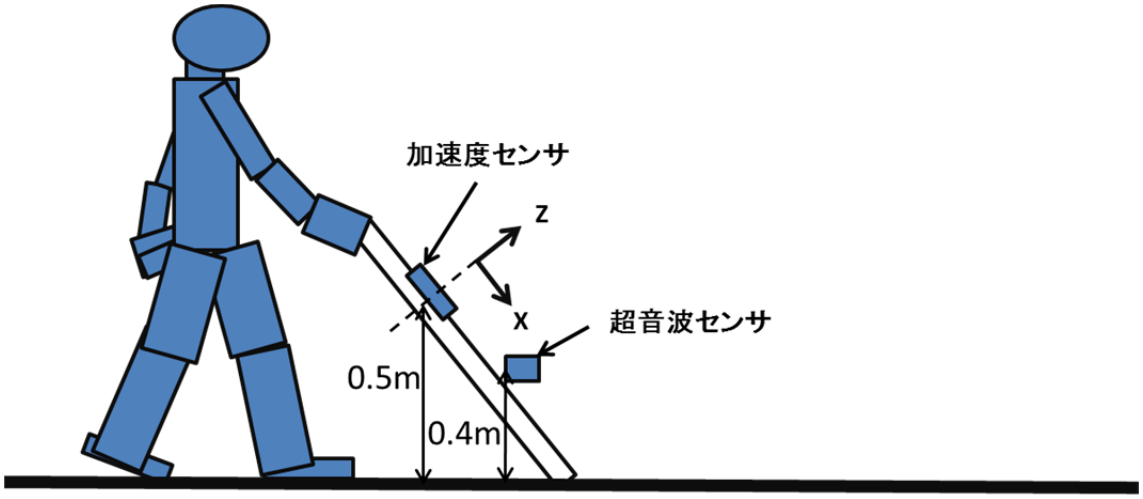


図 2.7　各種センサの取り付け位置

[6] 5 章参照

2.4　スマートフォン

1.1.4 で前述した通り，視覚障害者はスマートフォンの使用に関して積極的である．スマートフォンの操作にはタッチパネルなど視覚を用いた直観的な操作が必要であり，視覚障害者の方の使用は難しいと一般的に考えられる．しかし，各通信キャリアの支援や補助周辺機器などの発達により，視覚障害者の方に使用しやすい環境が整いつつある．その一例として NTT docomo から発売されている「らくらくスマートフォン」が挙げられる．これは，約 10 年以上にわたり現在も販売されている「らくらく」シリーズを Android 版にしたものであり，操作を簡略化し，見易さ，使いやすさを充実させた商品である．この製品の特長として,「触れる」と「押す」の違いを区別した新構造タッチパネルによる操作があり，画面を触れることにより選択しているものを音声で確認でき，押すことにより決定できる [19]．音声による文字入力や端末の操作などもできるなど，画面を確認できない視覚障害者にとっても有用であると考える．また， 1.1.3 で述べたように，視覚障害者向けのアプリなども開発されている．これらはスマートフォンが有する多機能性や Android OS の拡張性の高さにより実現できることである．

ここで，研究に使用する Android OS について触れておく．Android は，Google 社が中心になって組織された Open Handset Alliance によって開発されたプラットフォームであり，スマートフォンやタブレット PC などの情報端末を主なターゲットとしている．非常に汎用性が高いが，最も注目すべきは，誰でも無償で利用することができるオープンソースな OS であるということであり，これが Android OS の普及における大きな要因の 1 つであると考えられる．

我々はスマートフォンの携帯性と高い計算能力，Android OS の拡張性の高さに着目し，白杖のデータ処理部として利用を考える．これにより，スマートフォン（Android OS を搭載）を用いた新たな視覚障害者用電子補助機器分野が開拓できると考える．

2.5　白杖への実装

図 2.8 のようにそれぞれのセンサを白杖へ装着する．実装にあたり RT-ADKmini への給電には Panasonic 製の「無接点対応 USB モバイル電源 QE-PL102（モバイルブースター）」を用い，各センサへは RT-ADKmini から 3.3V の給電を行う．また，超音波センサは地面から 0.4m の位置，加速度センサは地面から 0.5m の位置にそれぞれ取り付けてある（図 2.7）．

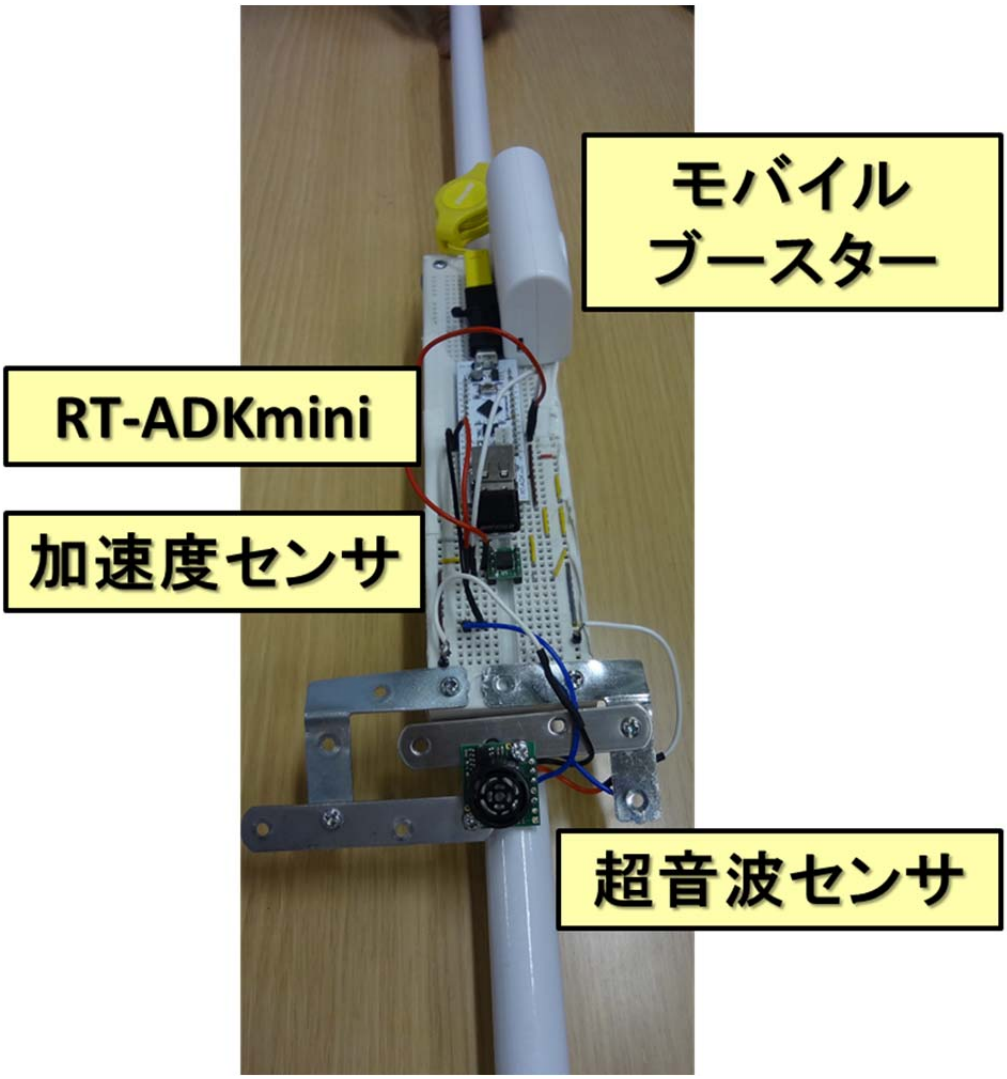


図 2.8　白杖への実装（試作版）

スマートフォンはポケットなどに入れたまま使用できるため，手に持つ必要はない．常時 Bluetooth 通信を用いてスマートフォンと通信を行っている（図 2.9）．

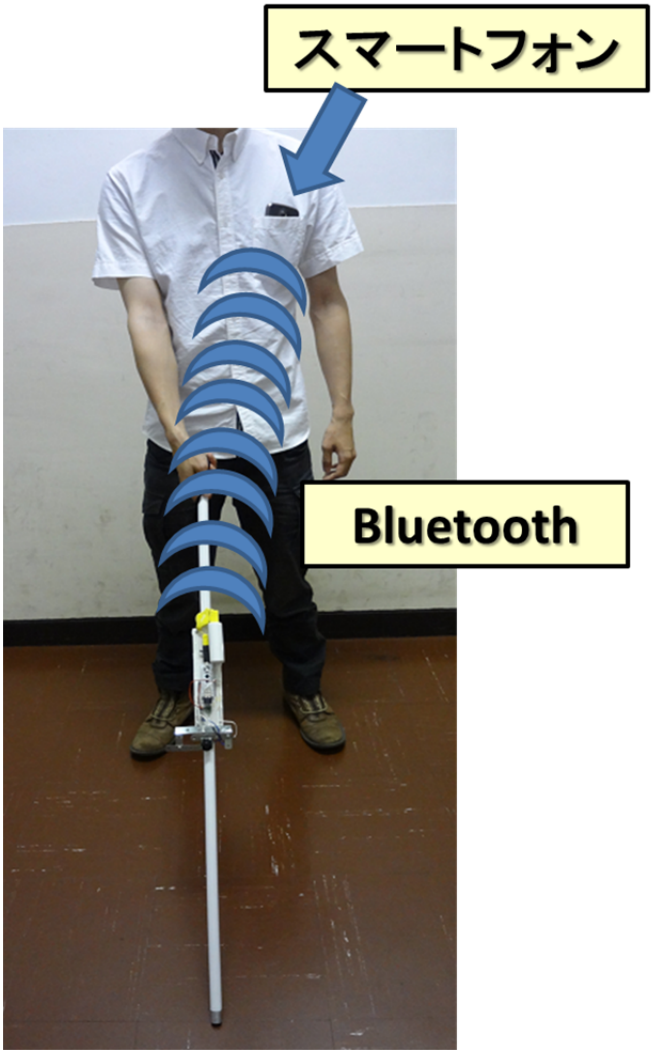


図 2.9　使用時風景

第3章　ファームウェアについて

本システムは各センサからのアナログデータを RT-ADKmini に搭載されている PIC24FJ64GB002（以下 PIC）を用いて A/D 変換を行って取得し，その値を Bluetooth 通信でスマートフォンへと送信している．本章では PIC で動作するファームウェア側の開発環境の構築からスマートフォンへのデータ送信までの説明を行う．

3.1　開発環境

PIC のプログラムを開発するためには開発環境ソフトがインストールされた PC，PIC にプログラムを書き込むための PIC ライタが最低限必要である．今回，開発環境ソフトにはマイクロチップ社が提供する MPLAB，PIC ライタには PICKit3 を使用し，開発言語は C 言語を用いた．MPLAB は図 3.1 のような構成になっている．これは総合開発環境であり，「プロジェクト」と呼ばれる開発単位で全てが総合管理されるようになっている．エディタ，アセンブラからシミュレータ，PIC ライタ制御，ICE 制御まで内蔵されており，アセンブラ言語で開発するときには，これだけで十分である [20]．C 言語を使用する場合には，C コンパイラが必須となり，MPLAB C30 をダウンロードし追加する必要がある．

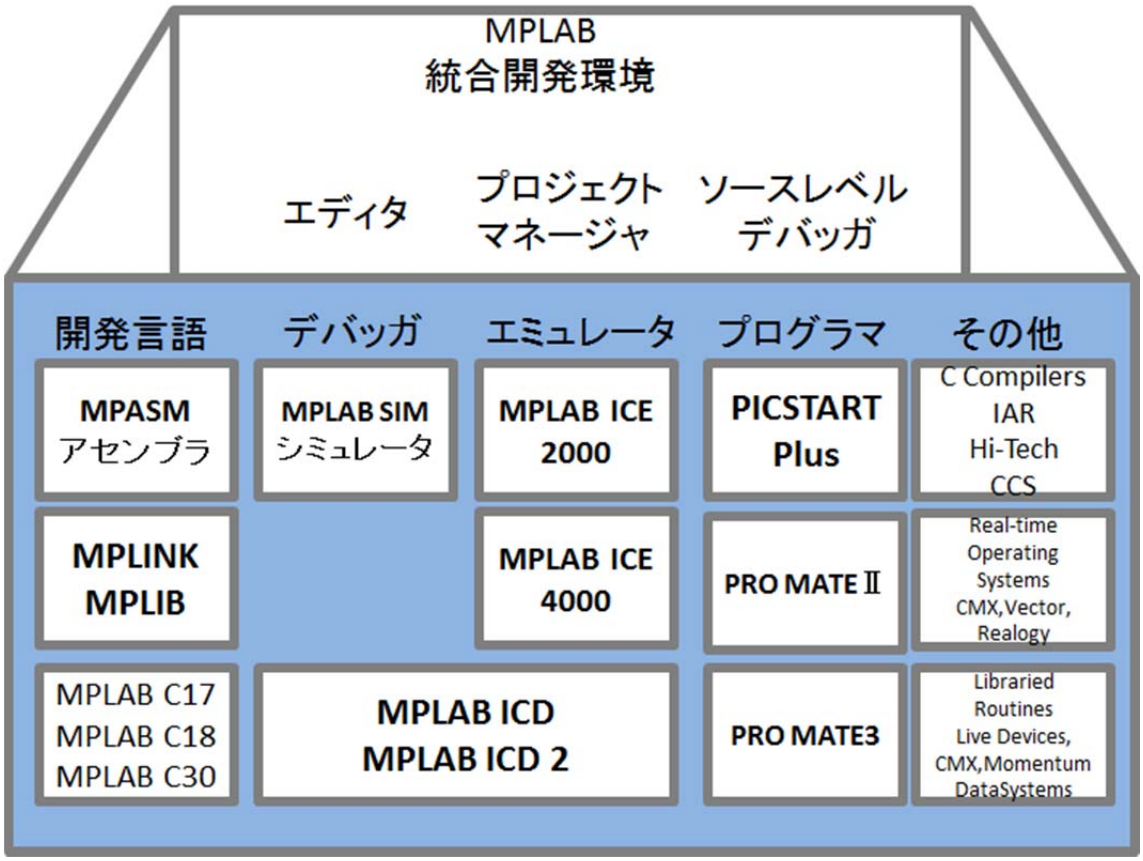


図 3.1　MPLAB の構成

3.2　A/D 変換

A/D 変換とはアナログ信号をデジタル信号に変換することである．加速度センサは単位時間の速度の変化率を電圧変化としてアナログ出力するため，PIC で処理するには A/D 変換でデジタル信号に変換を行わなければならない．また，超音波センサにはデジタル出力とアナログ出力があるが，今回は便宜上アナログ出力を用い，加速度センサのデータと同等に扱うことにする．PIC には図 3.2 のように 10 ビットの A/D コンバータが内蔵されている．このコンバータはサンプルホールドアンプも A/D コンバータ本体も 1 個だけなので，常に 1 入力ごとの A/D 変換となる．また，最大で 16 チャンネルの入力ピンがあり，上下両端のリファレンス電圧も外部入力可能だが，RT-ADKmini ボードが一部の Pin を利用していることで実際に使用できるのは 7 チャンネルまでとなる．付録 1 に回路図を添付する．以下では加速度センサの X，Y，Z 軸の 3 つと超音波センサ 1 つを含めた 4 チャンネルの A/D 変換をプログラムの一部を参照しながら詳細を示す [22].

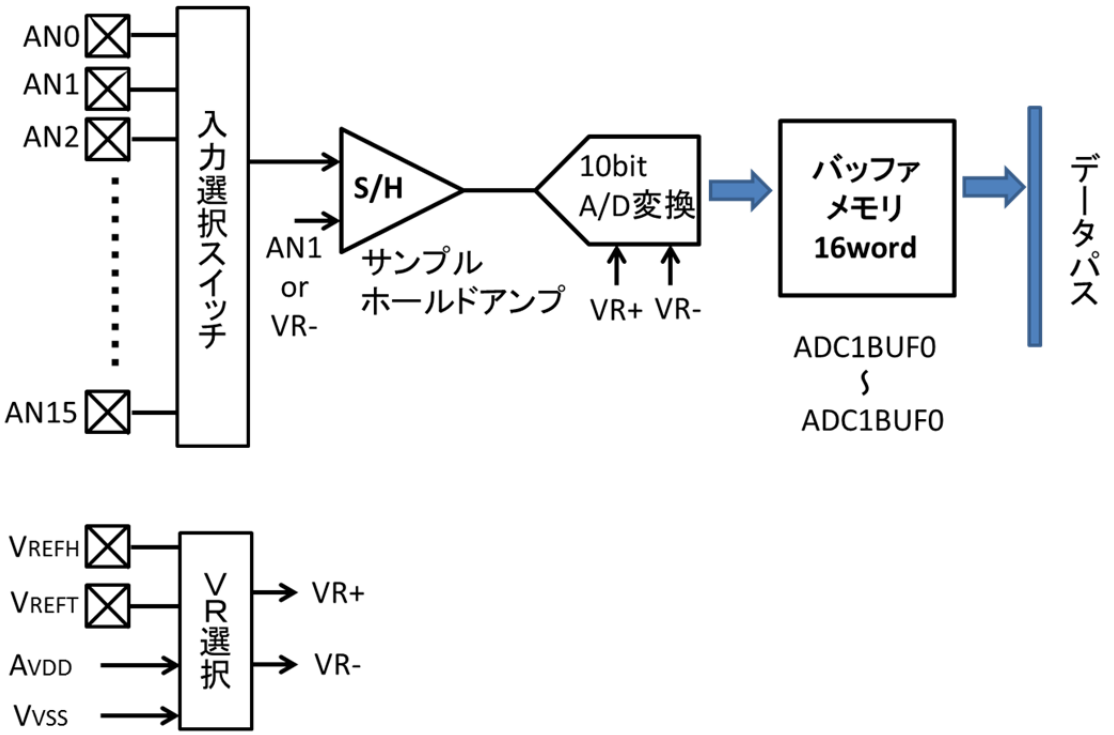


図 3.2　A/D 変換の構成 [22]

まず，A/D 変換で使用する変数を宣言する．

```
unsigned long acc_data_x;
unsigned long acc_data_y;
unsigned long acc_data_z;
unsigned long us_data;
//加速度センサ3つ、超音波センサ1つの宣言
```

次に，main 関数内で A/D コンバータの設定制御用のレジスタ設定を行う．以下にそれぞれの詳細を示す [23].

- AD1CON1：A/D 制御レジスタ 1．イネーブルと変換結果のデータフォーマット指定，サンプリング方法の設定
- AD1CON2：A/D 制御レジスタ 2．リファレンス電圧，使用チャンネル指定，バッファの使い方，割り込みのタイミング指定，自動スキャン指定などの設定
- AD1CON3：A/D 制御レジスタ 3．アクイジションタイムとクロックの設定
- AD1CHS：A/D チャンネル制御レジスタ．マルチプレクサ交互の切り替え設定
- AD1PCFG：A/D ポート構成レジスタ．デジタル入力-アナログ入力設定
- AD1CSSL：A/D スキャン選択レジスタ．自動スキャンの入力チャンネル設定

また，自動スキャン機能があり，アナログ入力を自動的に A/D 変換し，結果をバッファメモリに順番に格納し，指定チャンネル数だけ変換完了したら 1 回だけ割り込みを発生させる．その割り込みの設定が Timer3 となる．今回は 4 チャンネルの A/D 変換を行うため，自動スキャンには AN0，1，2，3 ピンを使用し，4 番目のサンプル/変換シーケンスの完了ごとに割り込みとしている．

```
//A/D変換
CLKDIV = 0;          //1:1クロック分周

///Timer3の設定 内部クロック 1/1 FOSC=16MHz
PR3 = 0xc35;                  //50msecインターバル
T3CON = 0x8000;

//ADCの設定
AD1CON1 = 0x8044;             //ADオン 整数 タイマートリガ サンプル自動
AD1CON2 = 0x040C;             //AVdd AVss 自動スキャン 4回目の割り込み
AD1CON3 = 0x1F05;             //31Tadサンプル ad =5*Tcy
AD1CHS = 0x0000;              //チャンネル選択(自動スキャン)
AD1PCFG = 0xFFF0;             //An0,1,2,3をアナログに
AD1CSSL = 0x000F;             //An0,1,2,3を自動スキャン

IEC0bits.AD1IE = 1;           //タイマーの割り込み許可
```

割り込み発生時にそれぞれの変数へ格納を行う．BUF0 から順番に加速度センサ X 軸，Y 軸，Z 軸，超音波センサに対応している．A/D コンバータは指定したチャンネルを若い番号から順番にスキャンしバッファメモリに格納していく．よって，PIC での入力ピンを間違うと任意の変数への格納が出来なくなってしまうので注意が必要である．

なお，それぞれのデータは 10 ビットの整数（0～1023）である．

```
void __attribute__((interrupt, auto_psv)) _ADC1Interrupt(void)
{
    IFS0bits.AD1IF = 0; //割り込みを受けたら、割り込みサービスルーチンの中でクリアする

    acc_data_x = ADC1BUF0;       //変数への格納
    acc_data_y = ADC1BUF1;
    acc_data_y = ADC1BUF2;
    us_data = ADC1BUF3;
```

3.3　スマートフォンへの取得データ送信

A/D 変換によりデジタル値となった取得データは，PIC から Bluetooth 通信を用いてスマートフォンへと適宜送信している．Bluetooth 通信とは近距離無線通信に特化した通信規格であり，現在ではこの機能を有するモジュールなどが一般に流通している．PIC で Bluetooth 通信を行うためには Bluetooth ドングルを使用するため，その通信のためのプログラムを PIC に書き込む必要があるが，ライブラリやプロファイル[7]を調べるなど膨大な作業となってしまう．そこで，PIC とスマートフォンとの間で Bluetooth 通信を行うため，WEB 上で無償公開されている hrdakinori 氏による BT_DROID を活用した [14]．デジタル値にした各センサデータを Bluetooth 経由でスマートフォンへ送信するための BT_DROID の変更点を挙げる．なお，スマートフォン側のシステムについては次章で説明する．

まず，bt_spp.c 内に extern 宣言を用い変数を宣言する．変数の実体は main.c 内部に存在する．

```
extern unsigned long acc_data_x;     //加速度センサx軸
extern unsigned long acc_data_y;     //y軸
extern unsigned long acc_data_z;     //z軸
extern unsigned long us_data;        //超音波センサ
```

次に，スマートフォンから PIC への通信として文字列での送信を行う．スマートフォン側から r で始まる文字列が送られ，PIC がそれを受信するとセンサ情報を文字列として送り返す流れになっている．既に述べたように 1 つのデータは 10 ビット（0～1023）で表現されているが，これを 16 進法で記述すると 000～3FF となり，3 バイトの文字列が必要であることが分かる．送り返すデータのため，r という 1 文字の後に 3 バイトのデータが 4 つの計 13 バイトに終端文字の 1 バイト分を足し，14 個分のメモリを確保する．その後，各センサのデジタル値を 3 バイトずつの文字列にして，スマートフォンに送信している．

---

[7] 機器固有の通信手段（プロトコル）を製品の特性ごとに標準化したもの．Bluetooth ではどのような順番・タイミングで，どのような種類の情報を転送すべきか，という機器の使い方にあたる手順を共通化しておく必要がある．

```
    if(p->tot_len != 0)
    {
          //送られたデータをそのまま返す
//引数を変える時に間違わないよう変数に入れておく。13は最初のrと3バイトが4つで13
           int qnum=13;
          q = pbuf_alloc(PBUF_RAW,qnum, PBUF_RAM);//送り返すデータの個数分メモリを確保


       if( *data == 'r'){
          char buf[14];

          //r、3ケタの16進数が4つ並んでいる。そこに入るのが後ろの4つ
          sprintf(buf,"r%3x%3x%3x%3x",(unsigned int)acc_data_x,
                                                (unsigned int)acc_data_y,
                                                (unsigned int)swit_data,
                                                (unsigned int)us_data);

          for(i=0 ; i<qnum ; i++){
              ((u8_t*)q->payload)[i] = buf[i];
          }


       }
```

これによりファームウェア側での Bluetooth 通信の設定は終了である.

第4章　Android アプリケーション

前章までに整備した環境を用いて，障害物の存在する方向を特定するアプリケーションの構築を行う．白杖を振りながら歩行を行うことにより障害物方向を特定し，スマートフォン内の内部メモリに時系列データとして保存することを目標とする．なお，内部メモリへの保存は取得データをグラフとして確認するためであり，実際の使用時は保存する必要はない．また，実験中の確認や展示会での健常者への展示の目的でスマートフォンの画面上にも結果を表示させる．

作成したアプリケーションは白杖に実装した超音波センサと加速度センサから送信されたデータを処理し，評価値計算を行う．開発言語には Java を用い，サンプリング周波数は 10Hz とする．

4.1　フローチャート

図 4.1 にデータ処理部のフローチャートを示す．まず，PIC とのペアリングにより Bluetooth 通信が開始され，データ通信が行われる．この際，毎秒 10 回（10Hz）で“r”という文字を Bluetooth 通信経由で PIC に送信し，通信に問題が無ければ各センサのリアルタイムなデータが送られてくる．3.3 で述べた通り，送られてくるデータは一つの文字列として連結されているため，分解した上で文字から数字への変換が必要となる．数字に変換後，白杖が地面を突いた時の加速度における衝撃の除去（ジャンプ処理）になるが，これについては 4.2 で説明する．

以上の前処理の後に評価値計算を行う．評価値とは障害物の方向を決定する際の指標として用いるものであり，評価値の理論と計算方法などの詳細を第 5 章で説明する．

その後，実験中に確認を行うために，スマートフォンの画面に現在の評価値を表示させる．評価値計算により決定された評価値はある一定の閾値を設けることにより左右の判別ができる．閾値 1 と閾値 2（閾値 1＜0＜閾値 2）の絶対値は等しいとし，条件を満たした場合のみ障害物方向を決定し方向を表示させる．その後，内部メモリへの保存を行い，次の取得データ処理を開始となる．

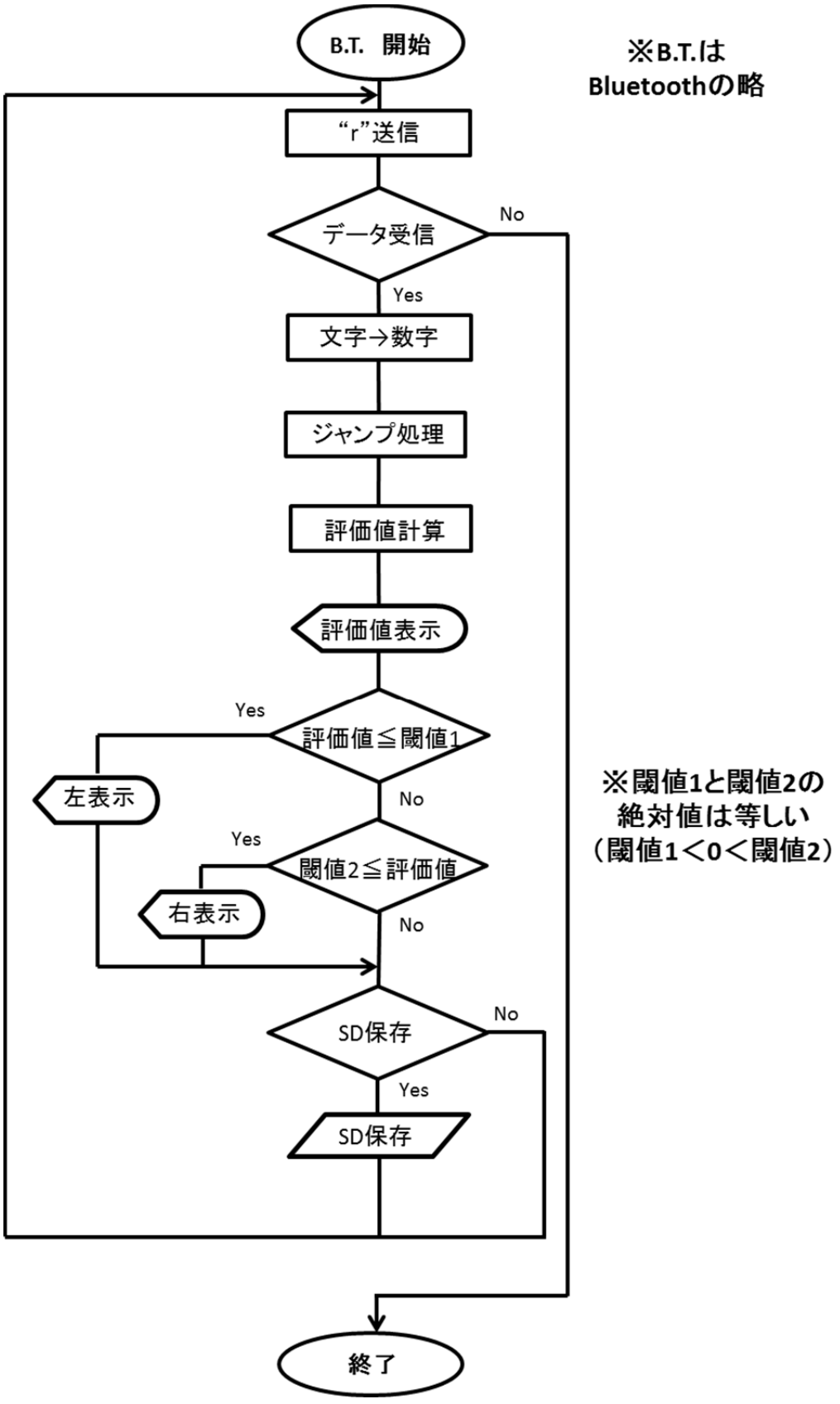


図 4.1　データ処理部フローチャート

4.2　衝撃の除去

白杖を用いた歩行方法には地面にスライドさせる「コンスタントコンタクトテクニック（スライドテクニック）」と離れた 2 点をタッチしながら歩く「タッチテクニック」とがある．スライドテクニックは地面の凹凸に敏感に反応でき，タッチテクニックではスライドテクニックより効率的に歩くことができる．どちらの歩行方法を用いるかは視覚障害者により変わるが，その環境や状況によって使い分けるのが理想とされている．白杖に取り付けた加速度センサは常に白杖の動きを電圧変化として取得しているため，地面をスライドさせるスライドテクニックを用いた場合には問題ないが，2 点間をタッチしながら歩くタッチテクニックを用いた場合では石突[8]が地面を突いた瞬間に生じる衝撃が急激な加速度として現れる場合がある（図 4.2 の赤丸内）．この衝撃が生じてしまうと，第 5 章で挙げる評価値計算の際に誤計算してしまう可能性が生じる．そこで，地面に突いた時の意図しない衝撃を除去するのがジャンプ処理となる．

ジャンプ処理では PIC から Bluetooth 通信経由で受け取った加速度を変数 acc に格納し，変数 prevacc には一つ前の加速度データを格納する．現在の加速度と一つ前の加速度の引き算を行い，任意の閾値 THRD で設定した値以上になった場合に一つ前のデータに置き換えることを行っている．

```
int THRD = 100;
if(Math.abs(acc-prevacc)<THRD || skipped){
    sdata = acc;
    skipped = false;
}else{ // large
    sdata = prevacc;
    skipped = true;
}

prevacc = acc;
```

このジャンプ処理を行うことにより，図 4.3 のように地面に接触した場合の意図しない衝撃が除去され，白杖の動きのみを検知できる．

8 白杖の先端，路面と接する部分

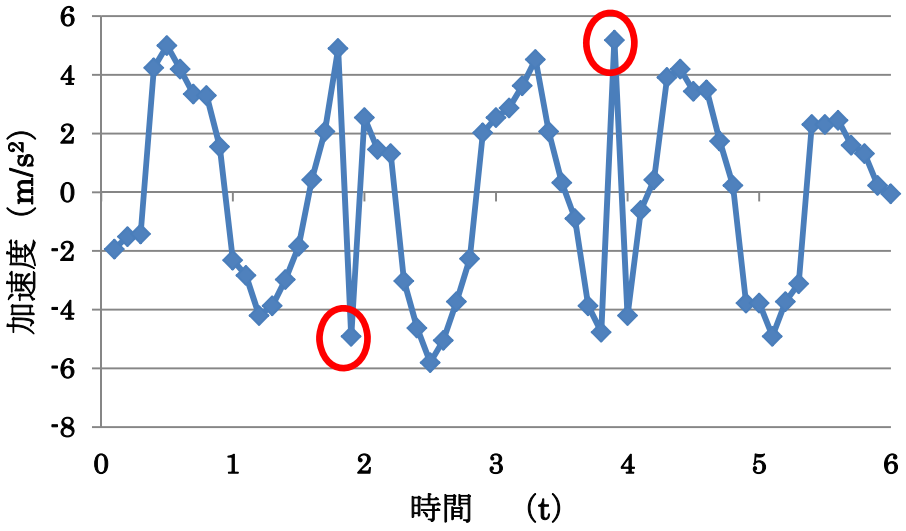


図 4.2　加速度センサの衝撃

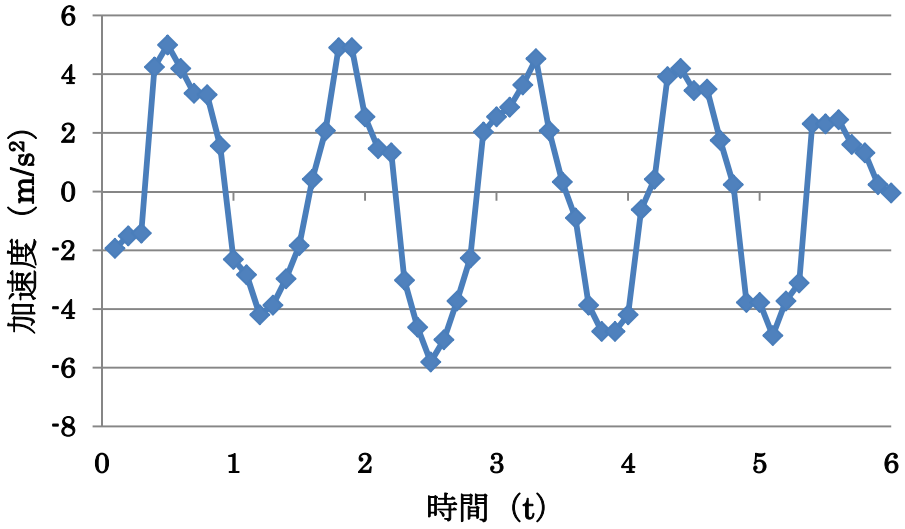


図 4.3　ジャンプ処理後

4.3　評価値計算の高速化

評価値計算を行うためには演算処理の高速化は必要不可欠となる．評価値計算には FFT や Newton 法などが含まれるため，処理が重くなり，動作にラグが生じて実用性に欠けると考えられるためである．本節ではプログラム内の評価値計算方法について説明し，評価値計算についての詳細は次章で説明を行う．

処理の高速化を行う手法としていくつかあげられるが，今回は JNI（Java Native Interface）を利用する．JNI とは Java で記述されたプログラムと他の言語（C，C++）で書かれたコードを連携するためのインターフェイスであり，計算量の多いプログラムを部分的にネイティブコードに置き換えて高速化する．本研究室の先行研究により，異なる方法でアプリケーションを記述し処理速度を比較すると，Java のみ（JIT[9]あり）での記述より，JNI（ユーザー関数の一部を C 言語化）を利用した場合，計算によっては約 5.6 倍処理速度が速くなることが分かっている [22].

以上の結果を基に，評価値計算部分には JNI（ユーザー関数の一部を C 言語化）を用いて記述を行った．Java 側には C 言語での処理結果を呼び出す以下の宣言を行う．

```
// JNIライブラリの宣言
static {
    System.loadLibrary("WhiteCaneSensors");
}
public native float getEvaluationValue(float time, float sdata, float sonic);
public native float getsdata();
```

実際に評価値計算の結果を呼び出すのが以下の部分となる．

```
float evaluation =getEvaluationValue(difftime, sdata, sonic);
```

この手法を用いることにより，計算量の多いプログラムを高い処理速度で実現でき，より有用な情報を抽出できる． C 言語でのプログラムソースに関しては付録 4 に JNI ソースコードに添付する．

[9] JIT(Just-In-Time Compiler)は Java で用いられる実行時コンパイルの総称. 中間コードから機械語への変換処理を実行直前にまとめて行う機能.

4.4　内部メモリへの保存

実験中に得られるデータを後に PC で確認するため, データをスマートフォンに保存する. 保存するデータは，PIC から Bluetooth 通信で送られてきた加速度センサ，超音波距離センサの取得データ，スマートフォン内部で演算を行った評価値である.

図 4.4 がアプリケーションの UI[10]となる．Bluetooth 通信が開始されると自動的にデータの取得と評価値演算が行われているが, 内部メモリへの保存はされていない. Start/Stop1 をチェックすると，センサの値を CSV ファイルとして内部メモリへの保存が開始される. SD カードへ保存されるデータは順番に，データ取得時間（difftime），加速度センサ X,Y,Z 軸の加速度データ（printMessage1,2,3），超音波センサの距離データ（printMessage4），ジャンプ処理後の Y 軸加速度データ（sdata），評価値（evaluation）とする．これにより，データを後に PC で確認することができる.

```
try{
    if(bw!=null){
        bw.write(difftime+","+printMessage1+","+printMessage2+","+printMessage3+","
                                +printMessage4+","+sdata+","+evaluation+"\n");
                                    }
 }catch(IOException e){
 }
```

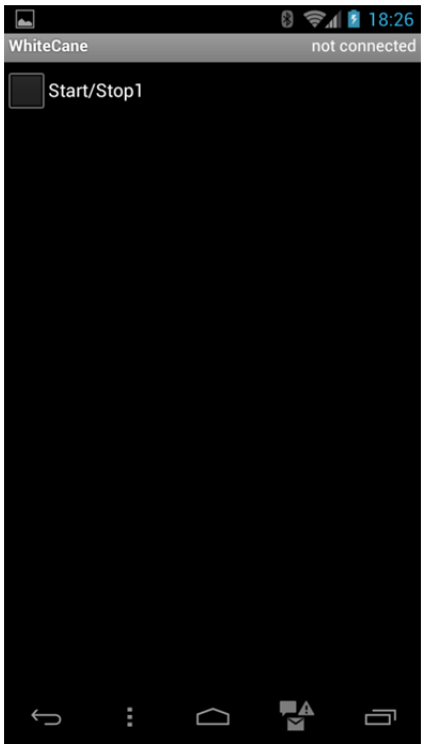


図 4.4　アプリケーションの UI

[10] ユーザーインターフェイス（User Interface）．機械と人間の間での情報のやり取りをするためのインターフェイス.

第5章　評価値についての理論

本章では評価値の理論，計算方法，障害物方向の判別について詳細を示す．評価値は加速度データと超音波センサからの距離データを用いて計算を行うもので，その結果を基に障害物方向を判定する．評価値を求める流れとして，

sin 波への近似（FFT による周波数決定　→　Newton 法による調整）→　評価値計算

となる．評価値計算は sin 波への近似を行った Y 軸方向の加速度値と超音波センサの距離の積を積分することにより算出される．

5.1　sin 波への近似

取得した加速度データは白杖の振り方により生じるノイズが含まれており，加速度の早い所や遅い所などばらつきが出てしまう（図 5.1）．そこで，白杖の使用方法である左右に振る行為に着目し，白杖の動き（加速度）を sin 波に近似することで，白杖の振り方から生じる誤差に影響を受けず障害物方向を認識できると考えた．

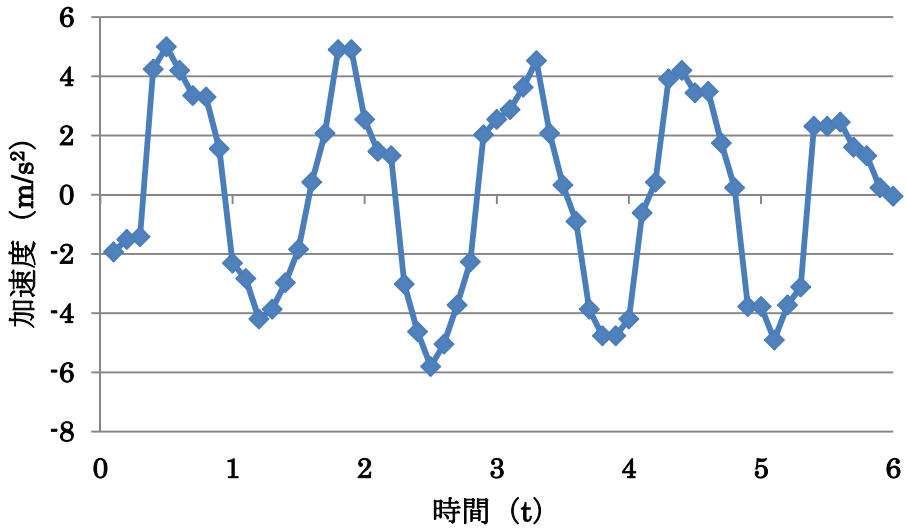


図 5.1　ジャンプ処理後の白杖の動き

sin 波を時刻 t[i]［秒］における信号値X(t)を sin 波と仮定すると（5.1）式となる．

$$X(t) = A\sin(2\pi f t[i] + \theta) \qquad (5.1)$$

ここでAは振幅，θ は位相を表す．加速度を（5.1）式で近似するために，周波数fの決定にはフーリエ変換を用い，時刻t[i]にはデータの取得時刻を用いる．また，実際の取得データとのずれを最小にするために振幅Aと位相 θ を Newton 法を用いて決定する．

5.1.1　FFT による周波数決定

周波数fを決定するためにフーリエ変換を用いる．サンプリングしたデータ自体が無限個あるとした場合，得られたフーリエ変換自体は周波数fに関する連続関数となる．しかし，実際にコンピュータ上で行う数値計算ではデータ数を無限とすることはできず，必ず有限個数となる．この場合，フーリエ変換は連続関数ではあり得ず，離散的な周波数に対する離散データとなり，これを計算するのが離散フーリエ変換（Discrete Fourier Transform：以下 DFT）である．一般に DFT を定義通りプログラムとして実装すると計算回数はN^2のオーダーとなり，非常に効率の悪い（時間のかかる）プログラムとなってしまう．その際，DFT を効率よく計算するアリゴリズムが高速フーリエ変換 FFT（Fast Fourier Transform：以下 FFT）となる [23].

sin 波への近似は取得データごとに行う．FFT を行う際，データ数 Nは必ず 2 のべき乗でなければならないため，本システムでは最新のデータ数から過去 32 個分までのデータを用いて周波数を決定する．データ数 32 個を用いたのは白杖の動きの約 2 周期分にあたり，誤差が少なく近似できるためである．以上により周波数fが決定する．

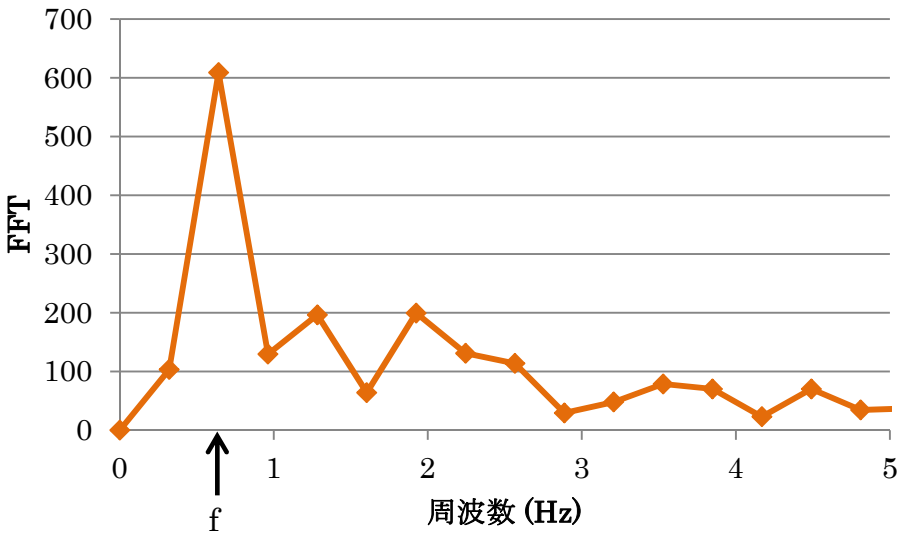


図 5.2　振幅スペクトル

5.1.2 Newton 法による調整

FFT により周波数を決定した後，振幅 A と位相 θ を最適化することにより，実測の加速度波形（以下加速度波形）と sin 波への近似波形（以下近似加速度）との一致を目指す．A と θ を決定するために, まずそれぞれの初期値を定めてから, Newton 法により調整を行う．Newton 法とは方程式の解に対する近似値を求めるための数値解析手法である．Newton 法で注意すべき点は，出発点に十分近い解を見つけることが出来れば非常に収束は早いが，初期値の選び方次第では収束をしなくなることである．例えば，解の初期値が真値から遠く，途中に関数の極値があるとうまくいかないことがある．これを解決するためには，解の近くの点に初期値を取るとよい．そのため，A はデータの標準偏差とし，θ は 0 から 2π を 32 分割して探索することで解に関しての大まかな当たりを付ける．その後，Newton 法を用いて A と θ の解に磨きをかけ決定する [24].

まず，データ取得開始からの経過時間を$\mathrm{t}[i]$，その時の加速度を$\mathrm{s}[i]$とすると，加速度波形と近似加速度の誤差を（5.2）式となる．これは加速度波形から近似加速度の値を減算し，二乗平均値を算出することにより誤差を導いている．

$$\mathrm{L} = \frac{1}{N}\sum_{i=0}^{N}\{s[i] - A\sin(2\pi f t[i] + \theta)\}^2 \tag{5.2}$$

次に，この関数 L を振幅 A, 位相 θ についての偏微分を行うことにより傾きを求め，F_1, F_2 と定める（(5.3)，(5.4) 式）.

$$F_1 = \frac{\partial \mathrm{L}}{\partial \mathrm{A}} = 0 \tag{5.3}$$

$$F_2 = \frac{\partial \mathrm{L}}{\partial \theta} = 0 \tag{5.4}$$

$$\mathbb{F}(A, \theta) = \begin{pmatrix} F_1 \\ F_2 \end{pmatrix} = 0 \tag{5.5}$$

（5.5）式を解くことにより， L の極値を求めることができる．その計算には，非線形連立方程式の Newton 法を用いるが，(5.5) 式で求められる値は極値であり，極大値も含んでしまう．しかし，適切な初期値を選ぶことにより極大値ではなく極小値を選択できる．L の極小値では加速度波形と近似加速度が最適に近似できる振幅 A と位相 θ が求められる．
ここで，F_1，F_2 についての偏微分の結果を以下に示す.

$$
\begin{aligned}
F_1 = \frac{\partial L}{\partial A} &= \frac{1}{N}\sum_i 2\left[\{s[i] - A\sin(2\pi f t[i] + \theta)\} \times \{-\sin(s\pi f t[i] + \theta)\}\right] \\
&= \frac{1}{N}\sum_i s[i]\sin(2\pi f t[i] + \theta) + A\frac{1}{N}\sum_i \sin^2(2\pi f t[i] + \theta) \\
&= -\left\{\cos\theta\frac{1}{N}\sum_i s[i]\sin(2\pi f t[i]) + \sin\theta\frac{1}{N}\sum_i s[i]\cos(2\pi f t[i])\right\} \\
&\quad + \mathrm{A}\left\{\cos^2\theta\frac{1}{N}\sum_i \sin^2(2\pi f t[i]) + \sin^2\theta\frac{1}{N}\sum_i \cos^2(2\pi f t[i])\right. \\
&\quad \left. + 2\sin\theta\cos\theta\frac{1}{N}\sum_i \sin(2\pi f t[i])\cos(2\pi f t[i])\right\}
\end{aligned} \tag{5.6}
$$

$$
\begin{aligned}
F_2 = \frac{\partial L}{\partial \theta} &= \frac{1}{N}\sum_i 2\left[\{s[i] - A\sin(2\pi f t[i] + \theta)\} \times \{-A\cos(2\pi f t[i] + \theta)\}\right] \\
&= -\mathrm{A}\frac{1}{N}\sum_i s[i]\cos(2\pi f t[i] + \theta) + A^2\frac{1}{N}\sum_i \sin(2\pi f t[i] + \theta)\cos(2\pi f t[i] + \theta) \\
&= -\mathrm{A}\left\{\cos\theta\frac{1}{N}\sum_i s[i]\cos(2\pi f t[i])\right. \\
&\quad \left. - \sin\theta\frac{1}{N}\sum_i s[i]\sin(2\pi f t[i])\right\} + A^2\sin\theta\cos\theta\left(-\frac{1}{N}\sum_i \sin^2(2\pi f t[i])\right. \\
&\quad \left. + \frac{1}{N}\sum_i \cos^2(2\pi f t[i])\right)(\cos^2\theta - \sin^2\theta)\left(\frac{1}{N}\sum_i \sin(2\pi f t[i])\cos(2\pi f t[i])\right)
\end{aligned} \tag{5.7}
$$

ここで Newton 法を多次元に適用するために（5.5）式を Taylor 級数に展開し，$\delta\mathbb{x}^2$以上の高次項を無視すると，関数値を同時にゼロに近づけるための補正$\delta\mathbb{x}$についての連立 1 次方程式

$$
\mathrm{A}\delta\mathbb{x} = \mathbb{b} \tag{5.8}
$$

が得られる．ここで，

$$
\delta\mathbb{x} = A^{-1}\mathbb{b} \quad , \quad \mathbb{b} = \begin{pmatrix} -F_1 \\ -F_2 \end{pmatrix} \tag{5.9}
$$

$$
\mathrm{A} = \begin{pmatrix} \dfrac{\partial F_1}{\partial A} & \dfrac{\partial F_1}{\partial \theta} \\ \dfrac{\partial F_2}{\partial A} & \dfrac{\partial F_2}{\partial \theta} \end{pmatrix} \tag{5.10}
$$

である．

（5.10）式の各成分は以下の通りである.

$$\frac{\partial F_1}{\partial \mathrm{A}} = \cos^2\theta \frac{1}{N}\sum \sin^2(2\pi f t[i]) + \sin^2\theta \frac{1}{N}\sum \cos^2(2\pi f t[i]) + 2\sin\theta\cos\theta \frac{1}{N}\sum \sin(2\pi f t[i])\cos(2\pi f t[i]) \tag{5.11}$$

$$\frac{\partial F_1}{\partial \theta} = -\left\{-\sin\theta \frac{1}{N}\sum_i s[i]\sin(2\pi f t[i]) + \cos\theta \frac{1}{N}\sum_i s[i]\cos(2\pi f t[i])\right\} + A\left\{-2\cos\theta\sin\theta \frac{1}{N}\sum_i \sin^2(2\pi f t[i]) + 2\sin\theta\cos\theta \frac{1}{N}\sum_i \cos^2(2\pi f t[i]) + (2\cos^2\theta - 2\sin^2\theta)\frac{1}{N}\sum \sin(2\pi f t[i])\cos(2\pi f t[i])\right\} \tag{5.12}$$

$$\frac{\partial F_2}{\partial \mathrm{A}} = -\left\{\cos\theta \frac{1}{N}\sum_i s[i]\cos(2\pi f t[i]) - \sin\theta \frac{1}{N}\sum_i s[i]\sin(2\pi f t[i])\right\} + 2A\left\{\sin\theta\cos\theta\left(-\frac{1}{N}\sum_i \sin^2(2\pi f t[i]) + \frac{1}{N}\sum_i \cos^2(2\pi f t[i])\right)(\cos^2\theta - \sin^2\theta)\left(\frac{1}{N}\sum_i \sin(2\pi f t[i])\cos(2\pi f t[i])\right)\right\} \tag{5.13}$$

$$\frac{\partial F_2}{\partial \theta} = -A\left\{-\sin\theta \frac{1}{N}\sum_i s[i]\cos(2\pi f t[i]) - \cos\theta \frac{1}{N}\sum_i s[i]\sin(2\pi f t[i])\right\} + A^2\left\{(\cos^2\theta - \sin^2\theta)\left(-\frac{1}{N}\sum_i \sin^2(2\pi f t[i]) + \frac{1}{N}\sum_i \cos^2(2\pi f t[i])\right)(-4\cos\theta\sin\theta)\left(\frac{1}{N}\sum_i \sin(2\pi f t[i])\cos(2\pi f t[i])\right)\right\} \tag{5.14}$$

(5.8)は LU 分解[11]により解くことができ，以上の計算結果を用いて得られた補正を(5.15)式の解ベクトルに代入する．

$$\mathrm{x}_i^{new} = \mathrm{x}_i^{old} + \delta\mathrm{x} \tag{5.15}$$

この過程を誤差が十分小さくなるまで繰り返すことにより誤差が少ない sin 波への近似ができる（図 5.3）．

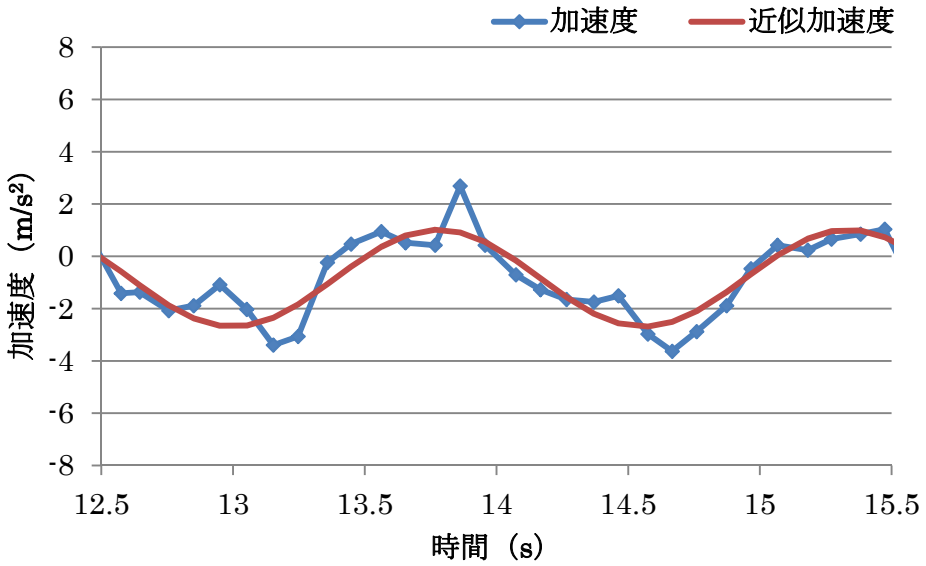


図 5.3　sin 波への近似

[11] 逆行列を計算するために使用．

5.2　評価値理論式

5.1 で述べた近似を行うことにより評価値の計算を行うことができる．評価値の計算では超音波センサからの距離データと sin 波へ近似したデータの二つを使用する．加速度センサの値は AD 変換により得られたものであるため常に正の値となるが，近似した加速度センサの値は加速度 0m/s² を基準に±に上下対称になる．この際，近似加速度のプラスの部分では白杖が左に振れている場合を示し，マイナスの部分では白杖を右に振っている場合を表している．

評価値を求めるには近似加速度の値を$G_0(t_i)$，距離の値を$G_1(t_i)$と定義し，その積の積分を行い算出する．

$$\int G_0(t)G_1(t)dt \approx \sum_i G_0(t_i)\,G_1(t_i)dt_i \tag{5.16}$$

積分を行うのは，ある一定の速さで振られている白杖の瞬間的な状態ではなく，近い過去の履歴により判断を下すためである．図 5.4 に示す通り，前方に障害物が存在しないまたは正面障害物までの距離が一定の場合には，積分を行っても近似加速度の±の積分値が同値となるため，評価値は 0 に近い値となる．

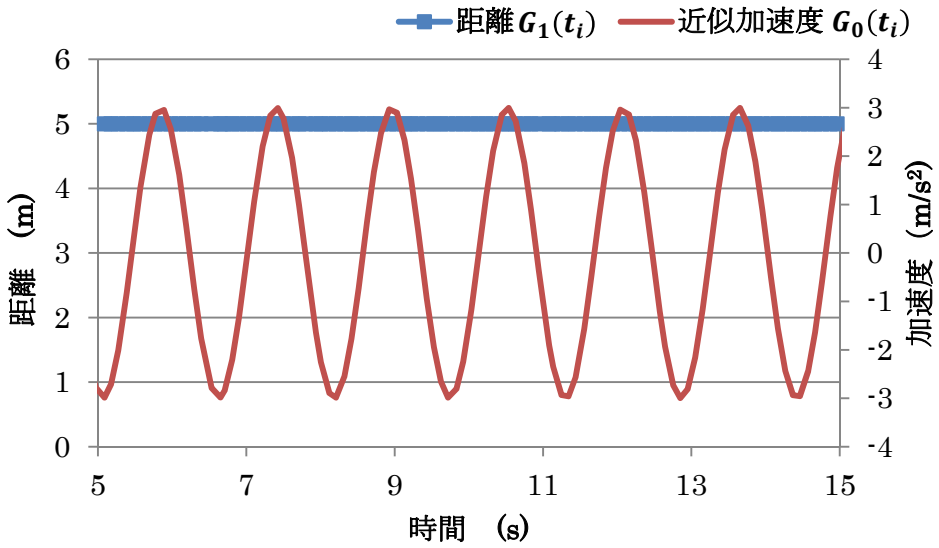


図 5.4　距離が一定の場合（評価値ほぼゼロ）

障害物が存在する場合には取得距離の値が変化することにより変動がみられる．右に障害物がある場合，プラスの加速度部分では距離が遠くなり，位相が揃うため積の積分値が大きいため評価値はプラスになる（図 5.5）． なお，図 5.5 と図 5.6 に示す評価値の実験環境については第 6 章で述べる．

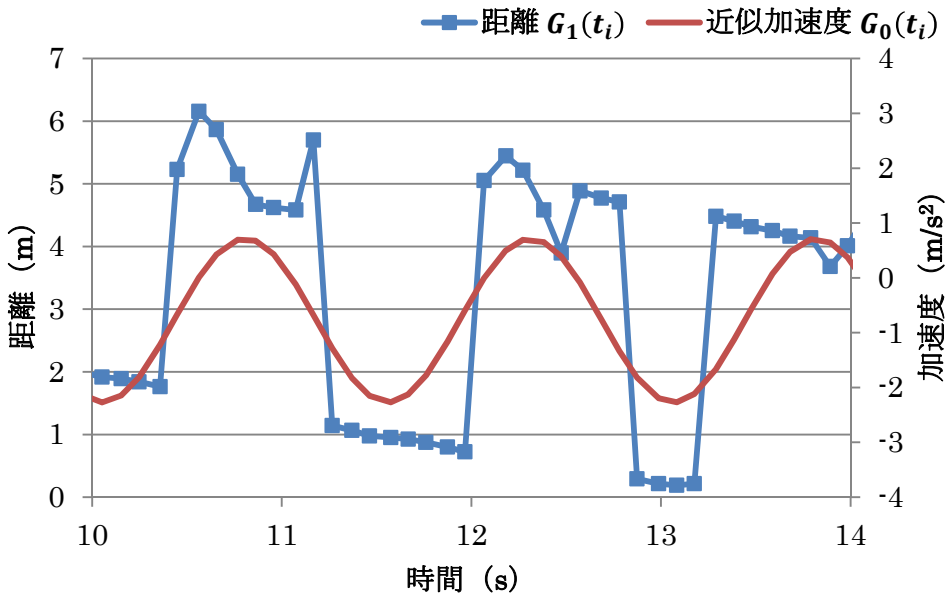


図 5.5　右方向に障害物（評価値プラス）

一方，左に障害物がある場合，プラスの加速度部分で距離の値が近くなり，位相がずれるため積の積分値が小さくなり，評価値はマイナスになる（図 5.6）.

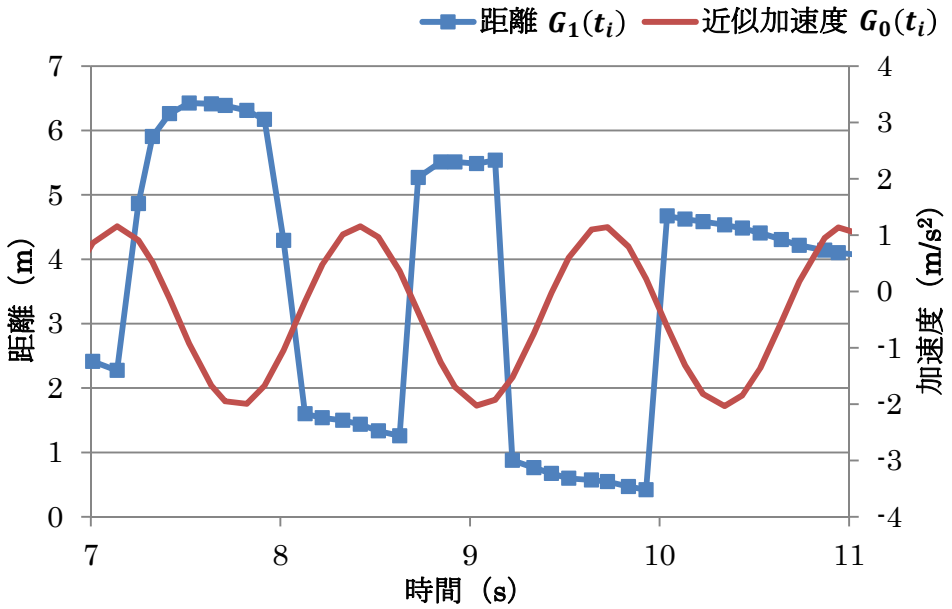


図 5.6　左方向に障害物（評価値マイナス）

上記の理論より評価値を算出している．評価値の基準と判別方法については次節に示す.

5.3　障害物方向の基準と判別

5.2 で述べた評価値計算を用いて障害物方向を決定する．評価値の値による場合分けにより，障害物方向を認識する．ここでは，実際の評価値計算結果を用いて評価値の基準について示す．なお，この実験の詳細については次章で説明する．障害物が存在しないまたは正面の障害物までの距離が同じ場合，評価値は図 5.7 2)のようになる．データ取得開始後の数秒の評価値が抜けているのは，Bluetooth の通信接続や sin 波への近似を行うためにデータ数を確保する待ち時間のためである．この結果から分かるように評価値が 0 付近で変動していることが確認できる．

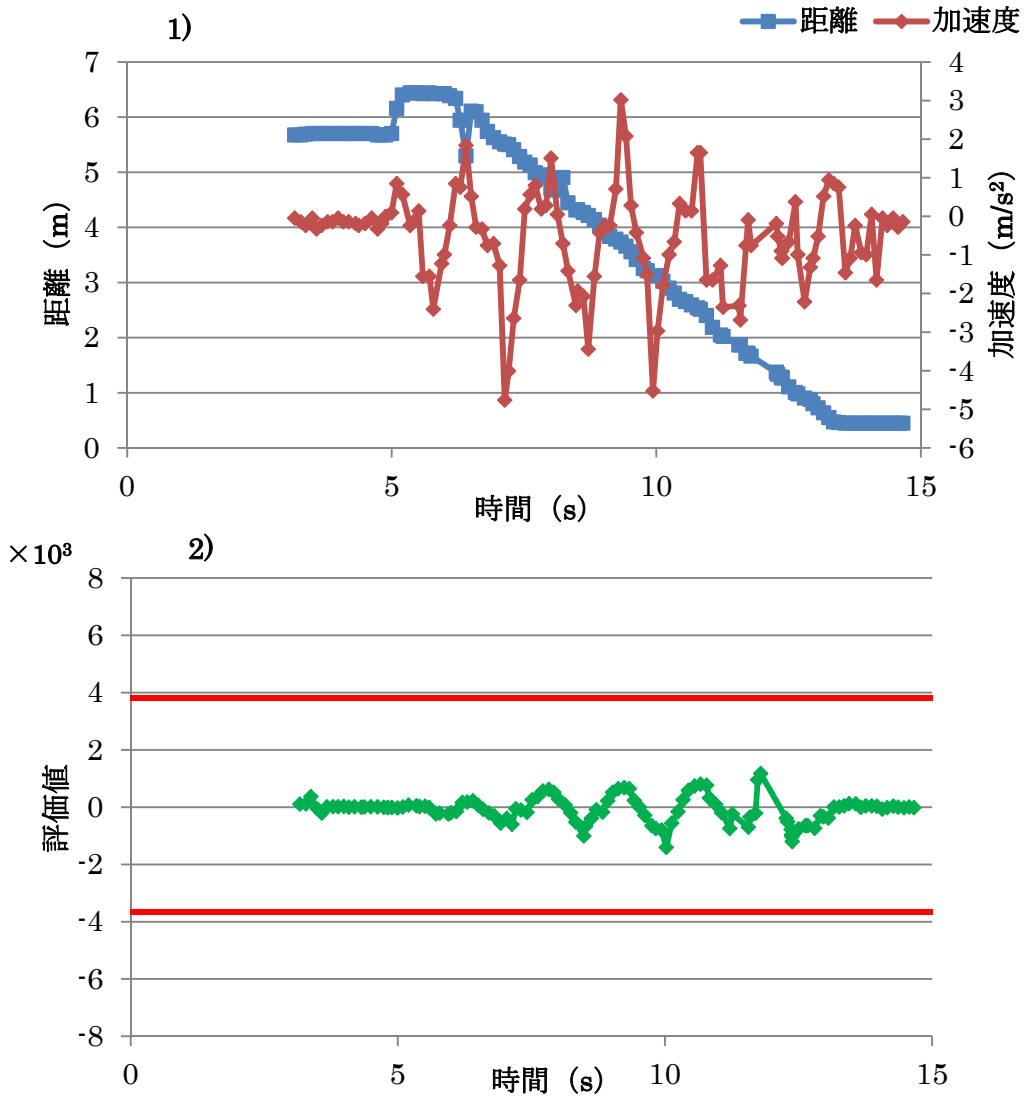


図 5.7　障害物なしの場合　1)距離-加速度グラフ　2)評価値計算結果

次に，右方向に障害物がある場合には図 5.8 のようになり，評価値はプラスになっていることが分かる．これは評価値理論で述べた通り，右に障害物がある場合，プラスの加速

度部分で距離が遠くなるため積分値が大きくなり，評価値はプラスとなるためである．

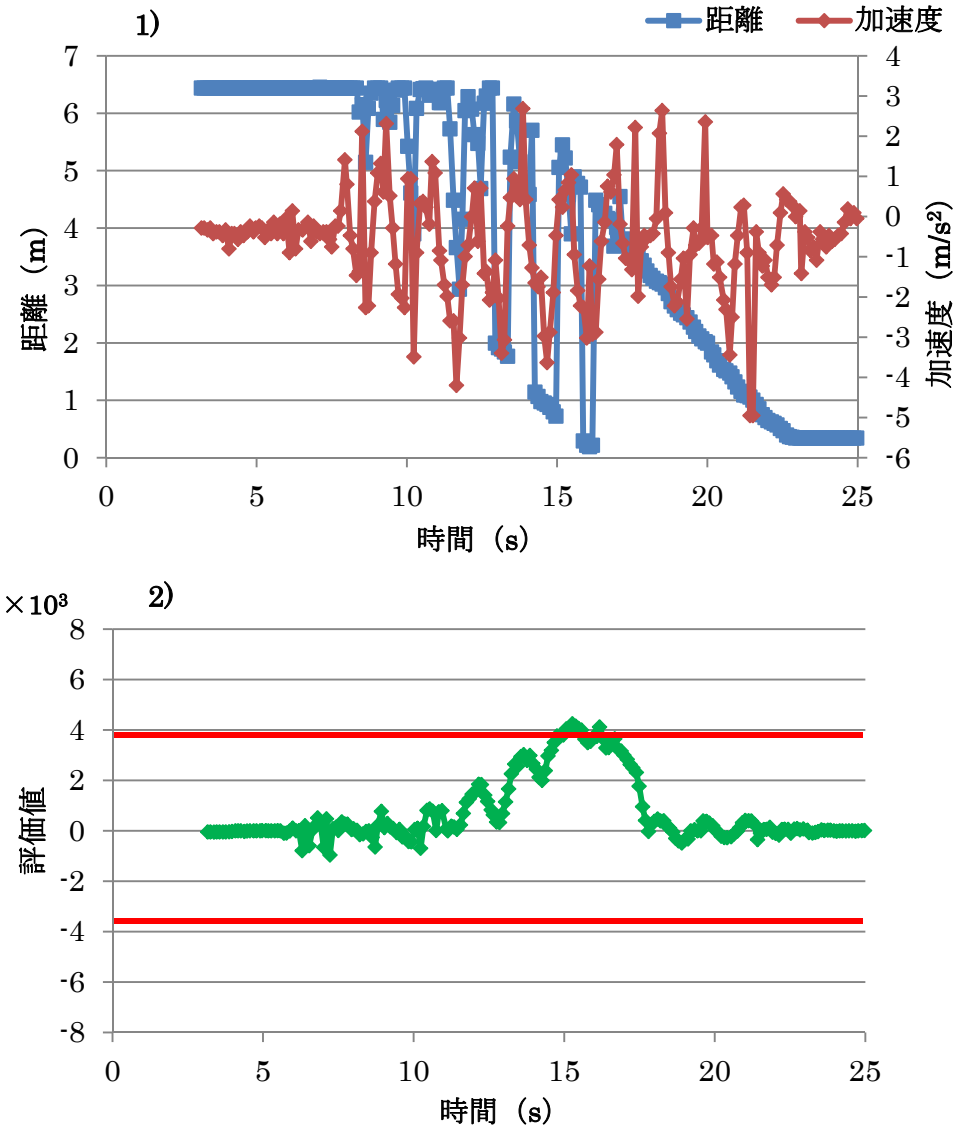


図 5.8　右方向に障害物ありの場合　1)距離-加速度グラフ　2)評価値計算結果

左方向に障害物のある場合の評価値グラフを図 5.9 1)に示す．右方向に障害物がある場合とは異なり，評価値がマイナスになっていることが確認できる．

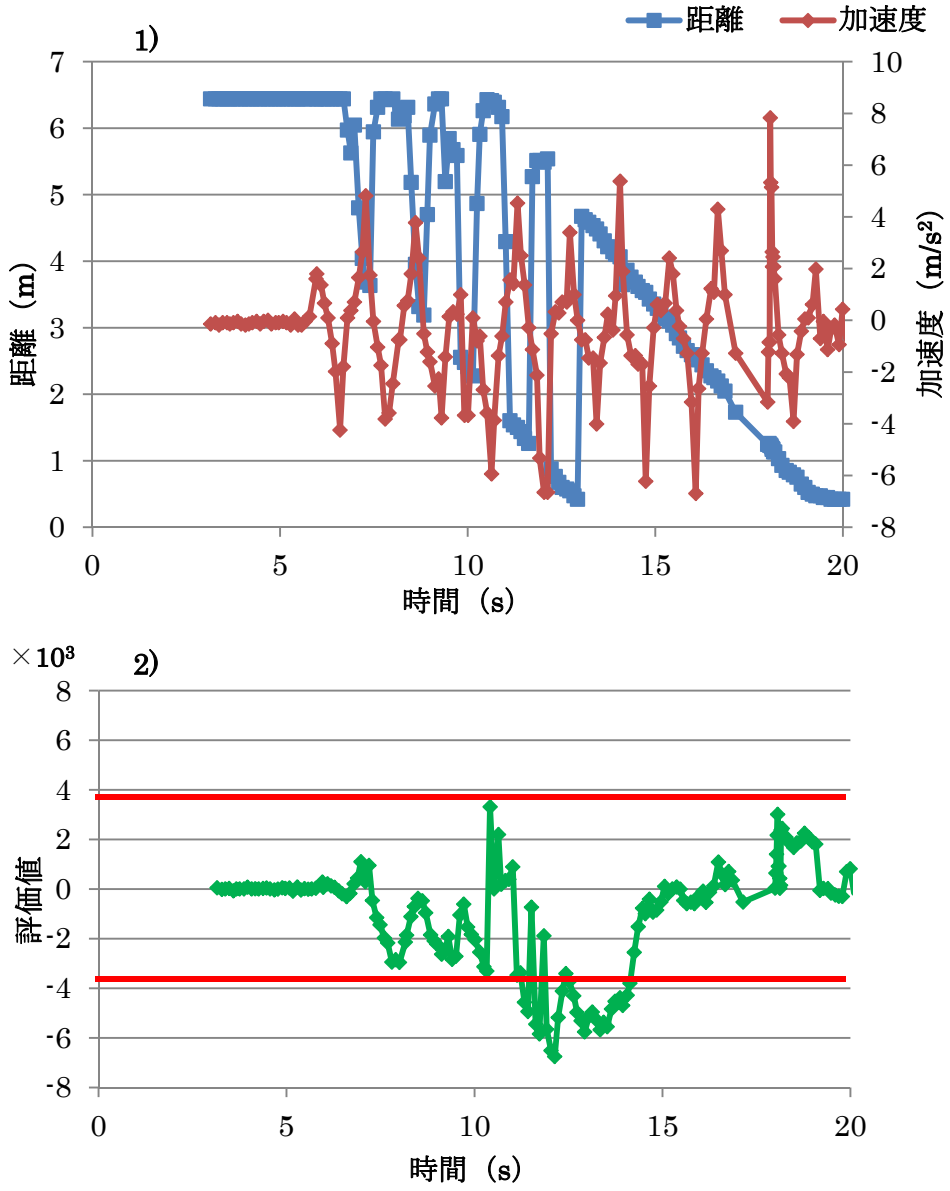


図 5.9　左方向に障害物ありの場合　1)距離-加速度グラフ　2)評価値計算結果

この 3 つの結果を踏まえ，評価値の場合分けを行うための閾値を設ける．閾値とはその値を超えることにより障害物がどちら方向にあるかを定める数である．図 5.7，図 5.8，図 5.9 においては評価値3.8×10^3の位置に引かれている赤い線がこの場合の閾値となり，評価値3.8×10^3を超えた場合には右方向，評価値-3.8×10^3を超えた場合には左方向に障害物があると判断できる．また，評価値$+3.8 \times 10^3$～-3.8×10^3の範囲に関しては，障害物方向を認識できない．この範囲に関しては，障害物が存在しない状態と正面に障害物がある場合のどちらも考えられるためである．この対策に関しては第 7 章で説明する．

評価値は本システムの使用環境により大きく変動するため，閾値の決定方法に関しては任意で決めている．評価値は 5.2 で述べた通り，距離データと加速度データの積を積分し求めるが，障害物方向までの取得距離と障害物が存在しない方向の取得距離の差が大きいほど評価値は大きくなるため，広い道や人の少ない所では評価値が大きくなる傾向がある．しかし，人混みなどのこの距離の差が大きく出にくい所では評価値が小さくなる傾向がある．よって，使用者の方がどのような環境で主に使用されるかによって閾値が変わるため，任意による決定とした．閾値は作成したアプリケーション内で変更可能としている（図 5.10）．シークバーを動かすことにより閾値を変更でき，その場に応じた適切な閾値を設けることができる．しかし，これでは視覚障害者の方が使用できないため，閾値決定の方法の改善が必要である．

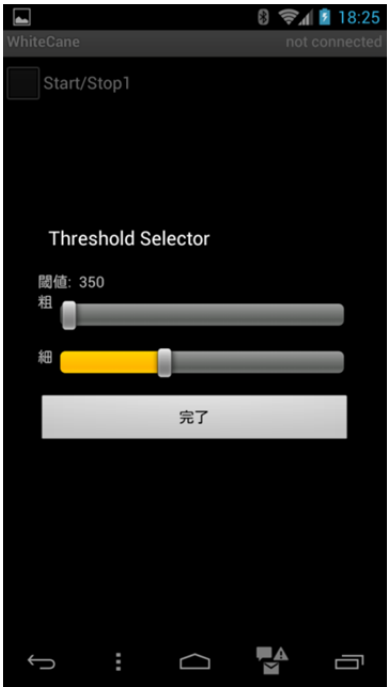


図 5.10　閾値の設定方法

以上，第 5 章で述べた計算をすべてスマートフォン内部で行い，障害物方向を特定している．

第6章　評価実験

6.1　　計測データ確認実験

2.5 で述べた超音波センサと加速度センサの実装を行った白杖を用いて，各計測データ取得状況の確認実験を行う．評価値計算を行う前に各センサの取得データ確認を行うことにより，加速度センサの白杖との相関と，超音波センサによる障害物までの距離の捉え方の確認を目指す．なお，評価値計算については 6.2 で示す．

6.1.1　　実験方法

使用機器と白杖への実装に関しては第 2 章で示したシステムを使用する．超音波センサは地面から 0.4m の位置，加速度センサは白杖に対して水平右向きを正方向とし，地面から 0.5m の位置に取り付ける（図 6.1）．加速度センサと超音波センサからのデータはサンプリング周波数 10Hz で取得し，スマートフォン内の内部メモリに CSV ファイルとして保存する．

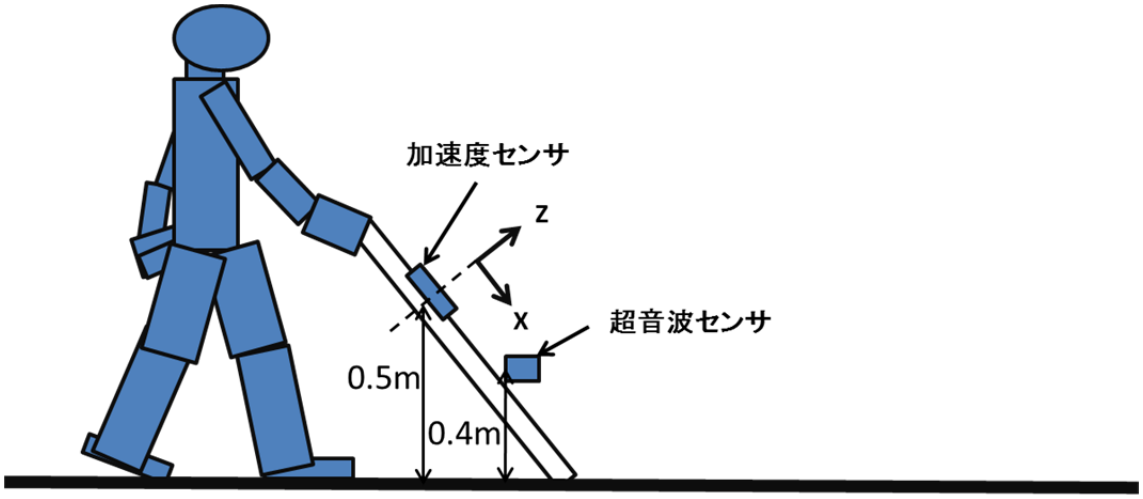


図 6.1　各センサの取り付け位置

実験環境を図 6.2 に示す．実験には周りに障害物の無い広い教室を使用し，壁から 7m 離れた地点から壁に向かって歩き，前方までの距離を取得する．その際，白杖は体の中心から左右 0.3mの範囲で左右に振りながら歩く．

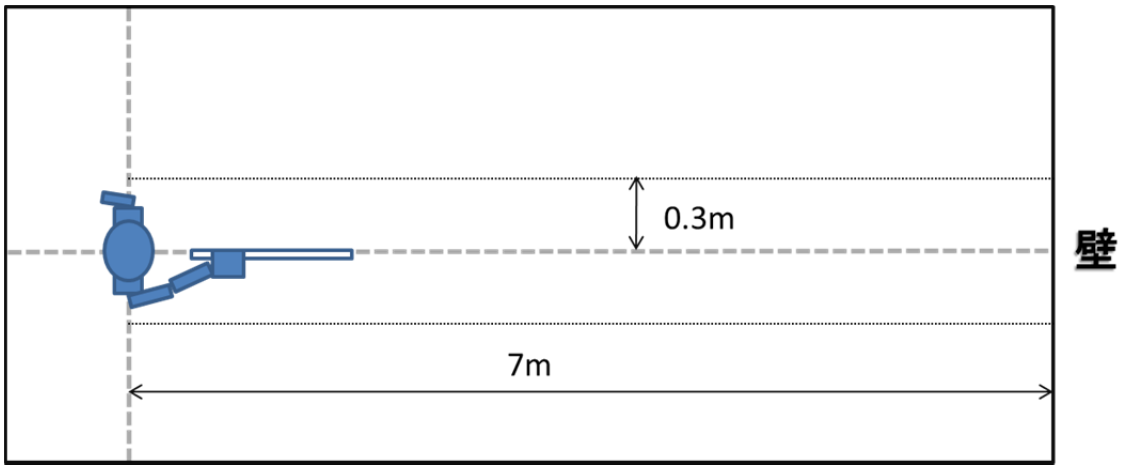


図 6.2　実験環境詳細

6.1.2　実験結果及び考察

スマートフォン内の内部メモリに CSV ファイルとして保存した加速度データと距離データの詳細についてそれぞれ示す．図 6.3 は PIC より受け取った Y 軸方向に対して加速度をジャンプ処理のみを行ったグラフであり，横軸はデータ保存開始からの経過時間とする．Y 軸方向の加速度を取得することにより白杖の振られている状態を認識できる．加速度 0m/s² を基準に±に変動することが確認できるが，プラスの部分では白杖が左に振れている場合を示し，マイナスの部分では白杖を右に振っている場合を表している．このグラフは評価値計算を行う際には第 5 章で述べた sin 波へ近似されるが，本質的な考え方は同じである．

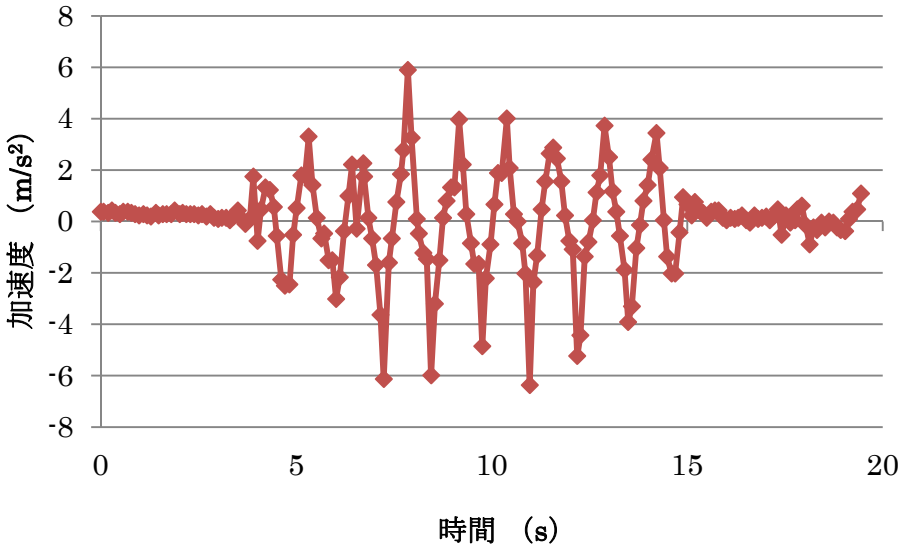


図 6.3　Y 軸方向の加速度変化

図 6.4 は超音波センサにおける障害物までの距離情報であり，横軸はデータ保存開始からの経過時間とする．本実験では 7m の位置から壁に向かって歩いているため，時間の経過とともに取得距離の値が減少していることが確認できる．これは，前方の壁までの距離を取得していることになる．また，データ保存開始時には距離検出限界を超えているため取得距離は最大値（6.5m）を示し安定している．しかし，3m から 6mの距離検出範囲において取得距離が不安定になっているころが確認できる．これは，白杖を振ることにより超音波センサが反射波を正常に受信できず生じた乱れだと考える．3m より近い部分で安定して距離が取得できたのは，壁までの距離が近くなったことにより，反射波のずれが少なくなったためである．本システムでは最長で 3mの範囲の障害物検知を目指すため，取得距離における若干の誤認識は影響しないとする．

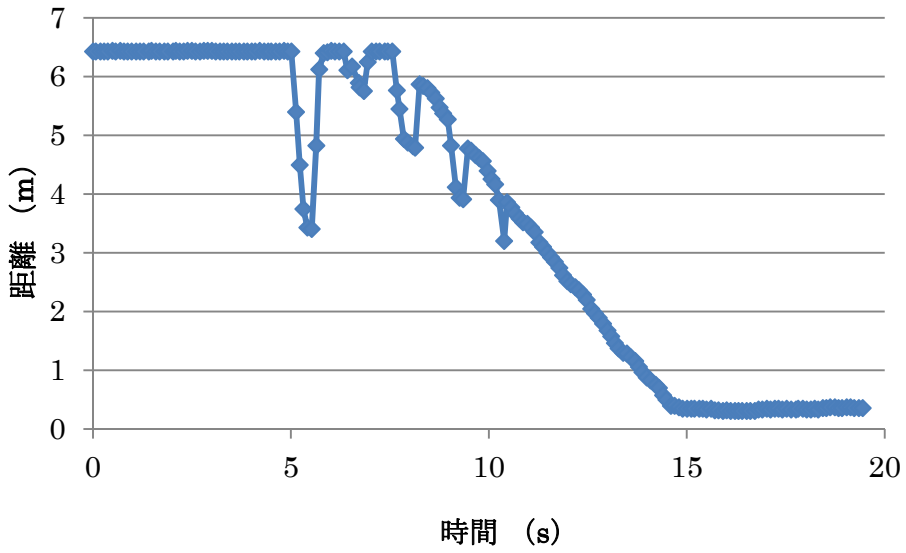


図 6.4　取得距離の変化

前述した二つのグラフを同時に表したものが図 6.5 である．このグラフより距離データに若干の誤認識は生じるものの，白杖の状態と壁までの距離を取得することができる．

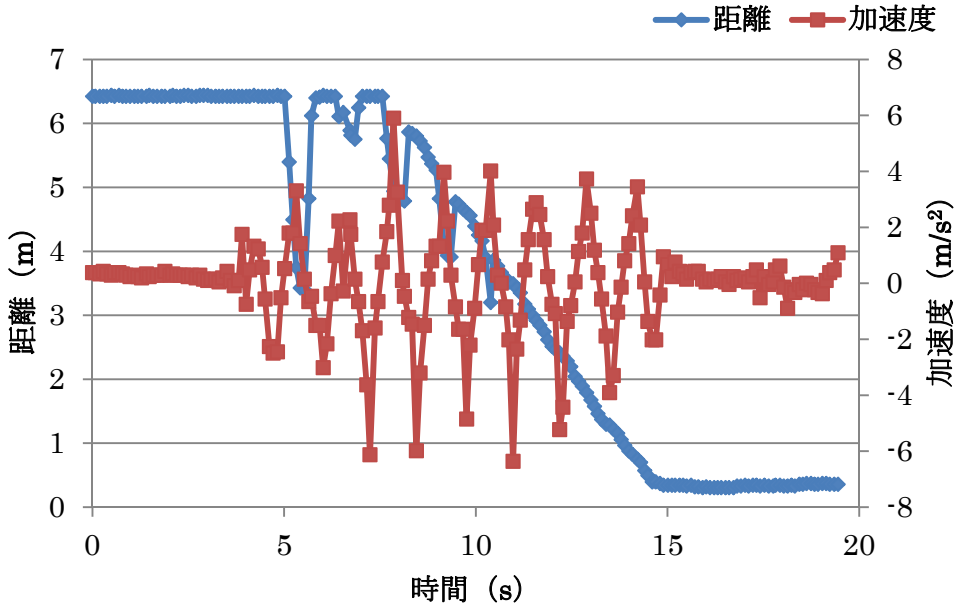


図 6.5　距離-加速度グラフ

6.2　評価値による障害物方向判別実験

6.1 で述べた二つのセンサを用いて，前方の障害物が使用者の左右どちらの方向にあるかを検知する実験を行う．障害物の方向判別には第 5 章で述べた評価値を用い，左方向と右方向，障害物なしの場合の 3 パターンの認識を目指す．

6.2.1　実験方法

使用機器，白杖への実装方法は 6.1.1 と同様とする．実験環境は実験には周りに障害物の無い広い教室を使用し，壁から 12m 離れた地点から壁に向かって歩き，前方の距離を取得する．その際，白杖は体の中心から左右 0.3mの範囲で左右に振りながら歩く．

障害物は中心から左 0.3m（図 6.6）と右 0.3m（図 6.7）の 2 パターンの位置に置き，白杖の検知範囲より外にある．障害物の大きさは 0.6×0.9×0.3m のプラスチックの箱を使用した．本実験ではこの 2 パターンに加え，障害物が存在しない場合を含めた 3 パターンについて実験を行う．

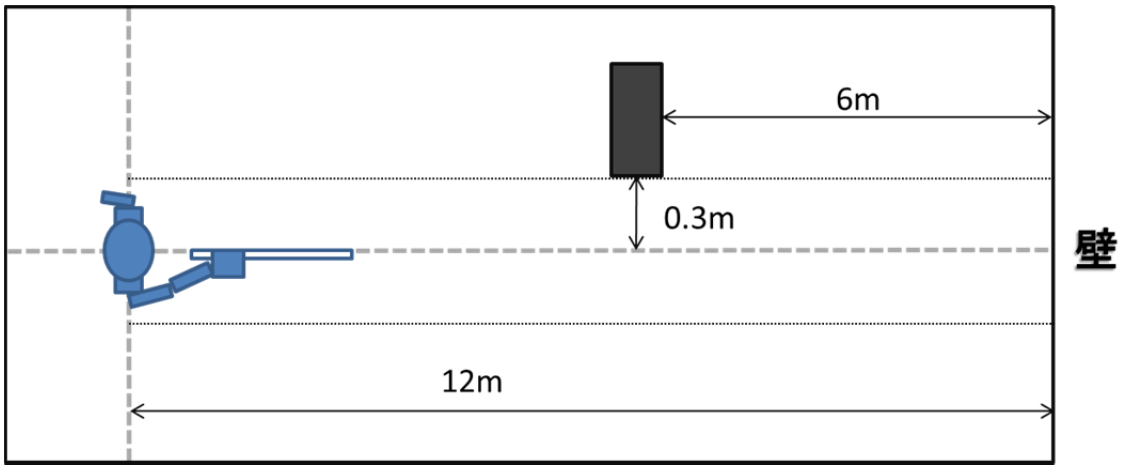


図 6.6　実験環境（左方向に障害物）

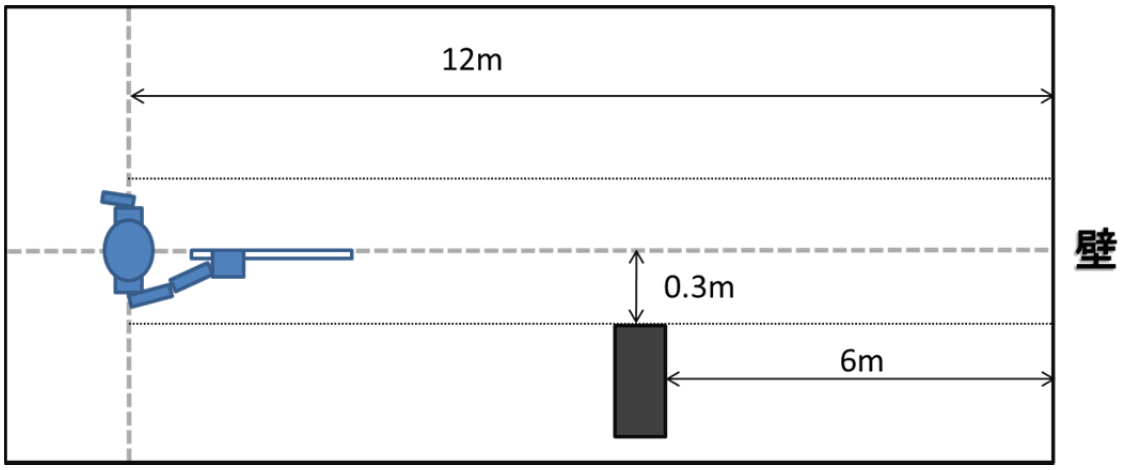


図 6.7　実験環境（右方向に障害物）

6.2.2　実験結果及び考察

まず，障害物が存在する場合の取得距離データの詳細を左 0.3m に障害物（図 6.6）を用いて示す（図 6.8）．実験者は壁から 12m 離れた位置から測定を開始するため，超音波センサの測定範囲内である 6m 先の障害物をまず検知する．障害物に超音波が当たるのは白杖が左に振られている間だけであり，正面と右方向においては障害物が存在しないことから測定限界の 6.5m を示す．そのため，取得距離のグラフには 5s～10s にかけて距離が 3m 以下に下がる所と 6m 付近への変動が生じる．10s の位置で障害物を通り過ぎた後，前方の壁までの距離が検知範囲内となり認識する．しかし，距離データだけでは障害物に近づいていることは分かるが，障害物の方向までは特定することができない．

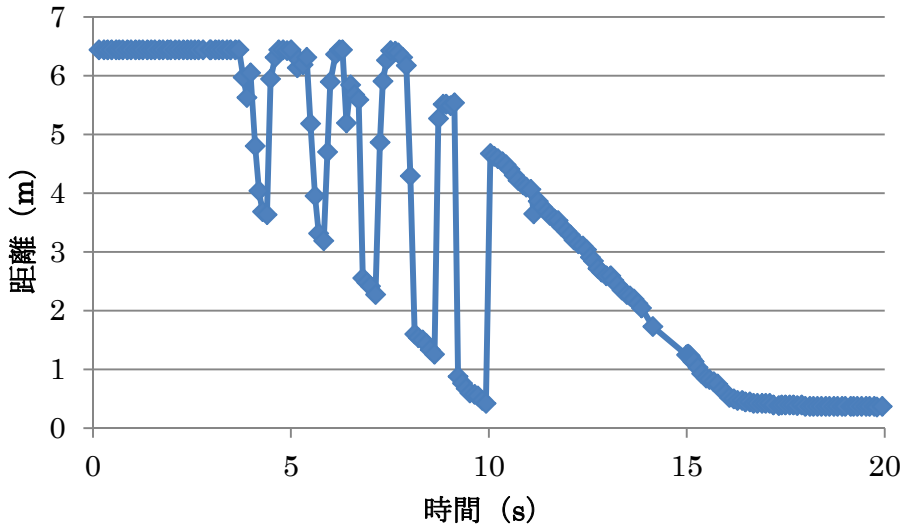


図 6.8　取得距離（左方向に障害物）

そこで，この取得距離と白杖の動きの相関を確認できるのが図 6.9 である．これは，ジャンプ処理を行った加速度による白杖の動きと障害物までの取得距離を同期させたグラフであり，3s から歩き出し，12m先の壁に到達するまで白杖を振っていることが分かる．図 6.10 はこのグラフの 5-10s 間を拡大したグラフとなる．7s 付近で加速度がマイナスになっている（緑枠内）のは意図しない衝撃によるもので，sin 波への近似を行うことにより評価値計算への影響はないと考える．

また，図 6.9 を近似加速度に変換し，3s～14s 間を拡大したものが図 6.11 1)となる．距離の値が小さくなる部分において，加速度がプラスになることが確認できる．これは白杖が左に振られている時に，障害物までの距離が近くなっていることを示している．

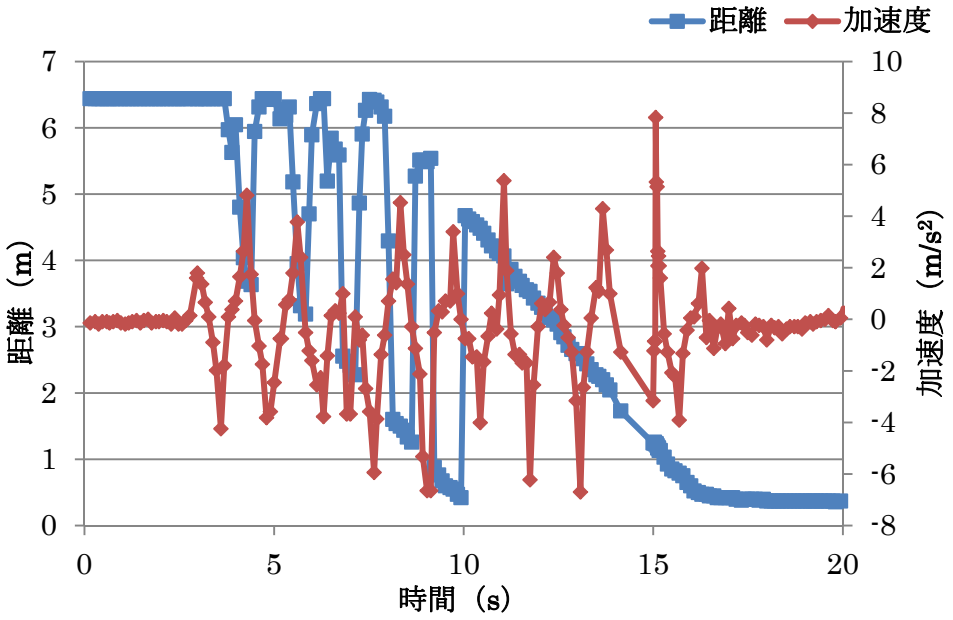


図 6.9　距離-加速度グラフ（左方向に障害物）

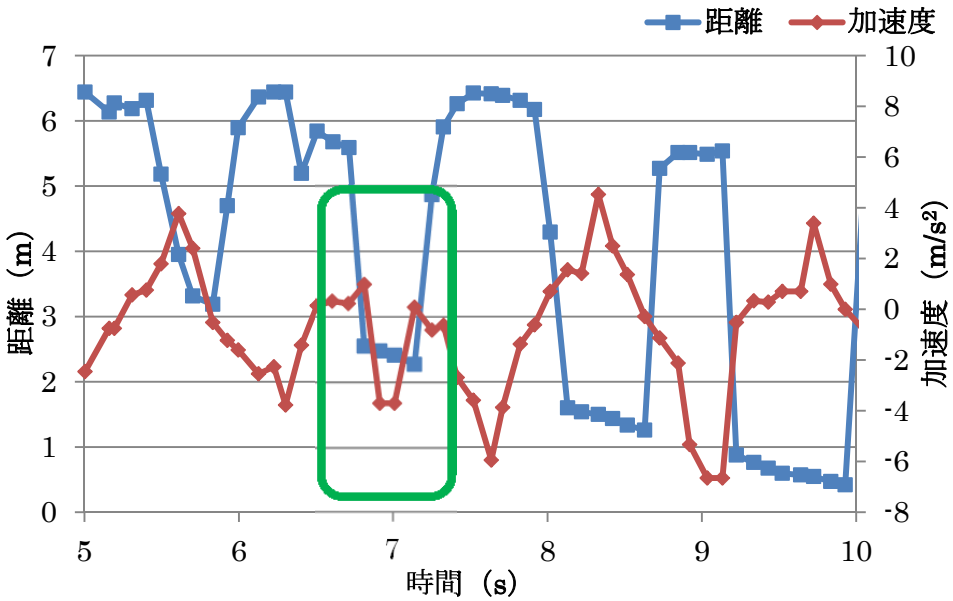


図 6.10　5-10s 間の意図しない衝撃部分

以上の実験を第 5 章で述べた評価値を用いて，取得データごとに計算を行った結果が図 6.11 2)である．障害物に最も近づく 10s まで評価値がマイナス方向に変動するのが確認できる．10s を過ぎた後も評価値がマイナスになるのは取得してから過去 32 個分のデータを用いて評価値計算を行うため過去の影響を受けているが，その後，評価値 0 付近に収束しているのが確認できる．また，8s 付近において評価値がプラスになっている部分があるが，障害物検知時の評価値との差が大きいため，閾値の設定により影響を受けない．

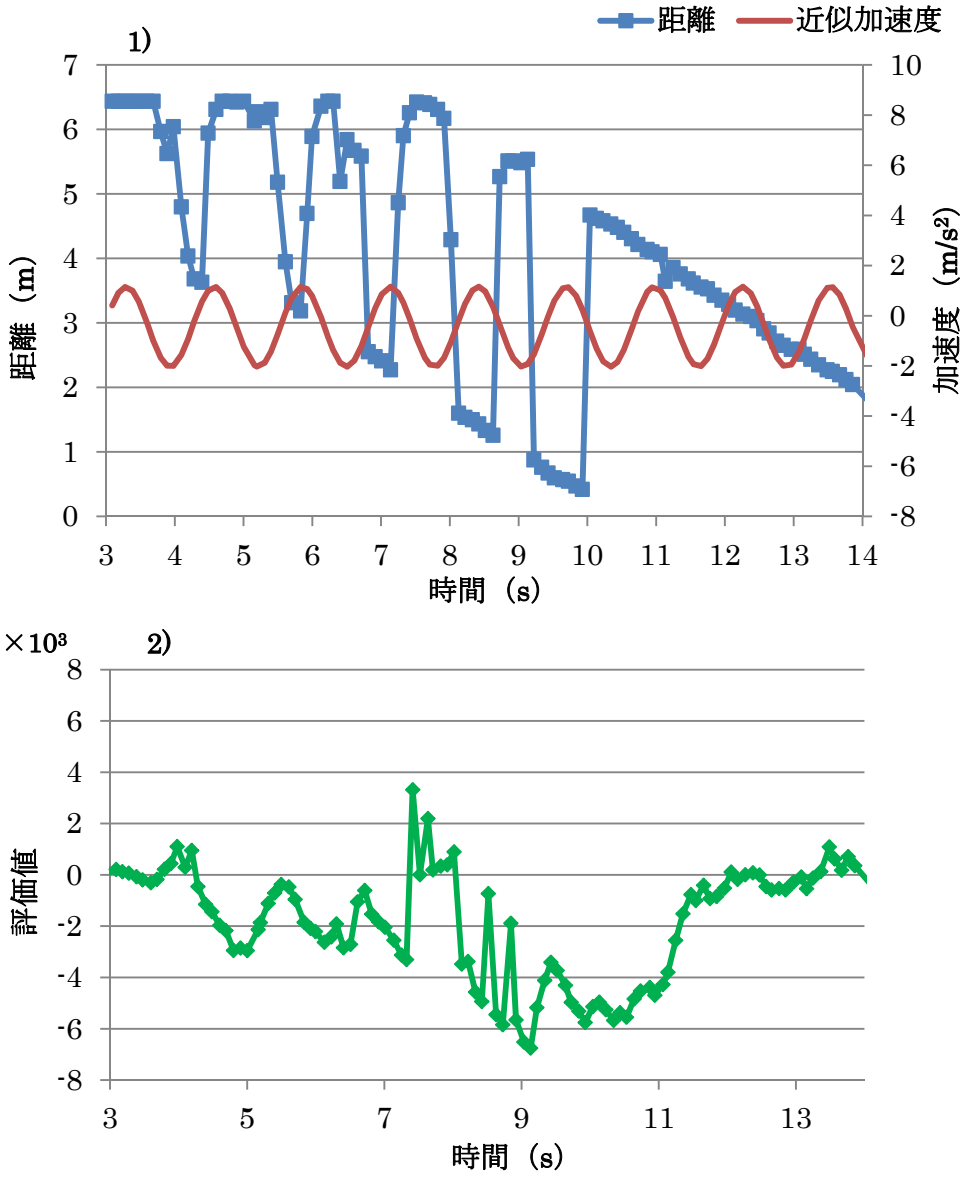


図 6.11　左方向に障害物　1)3-14s 間の詳細　2)評価値計算結果

次に，障害物が右 0.3m の位置にある場合（図 6.7）について示す．この場合の距離-加速度グラフは図 6.12 となり，加速度を近似加速度へ変換し，4s～16s 間を拡大したものが図 6.13 1)となる．左方向に障害物のある場合とは異なり，距離の値が小さくなる部分において，加速度がマイナスになることが確認できる．これは白杖が右に振られている時に，障害物までの距離が近くなっていることを示している．

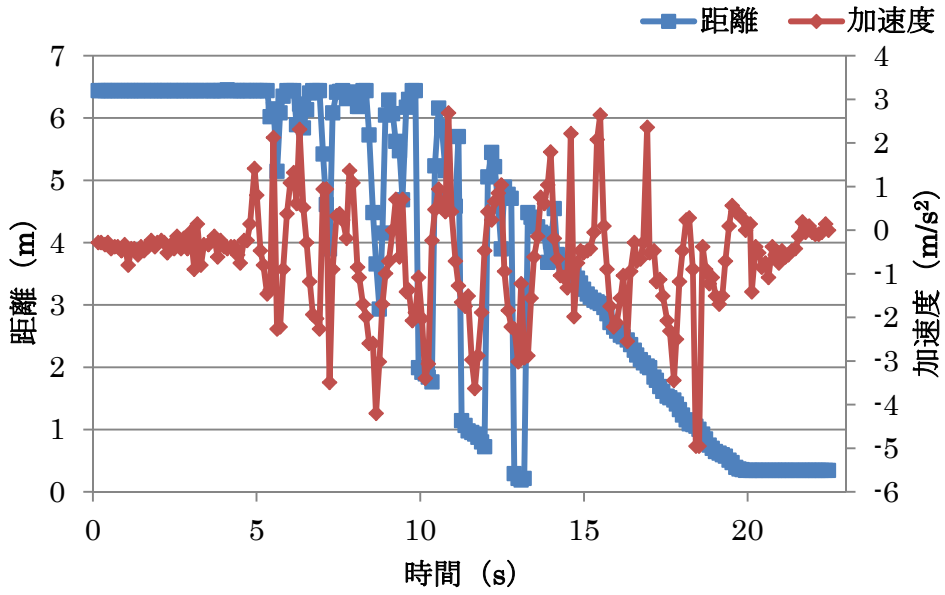


図 6.12　距離-加速度グラフ（右方向に障害物）

右方向に障害物のある場合において，評価値計算を行った結果が図 6.13 2)である．障害物に最も近づく 13s 付近まで評価値がプラス方向に変動しているのが確認できる．

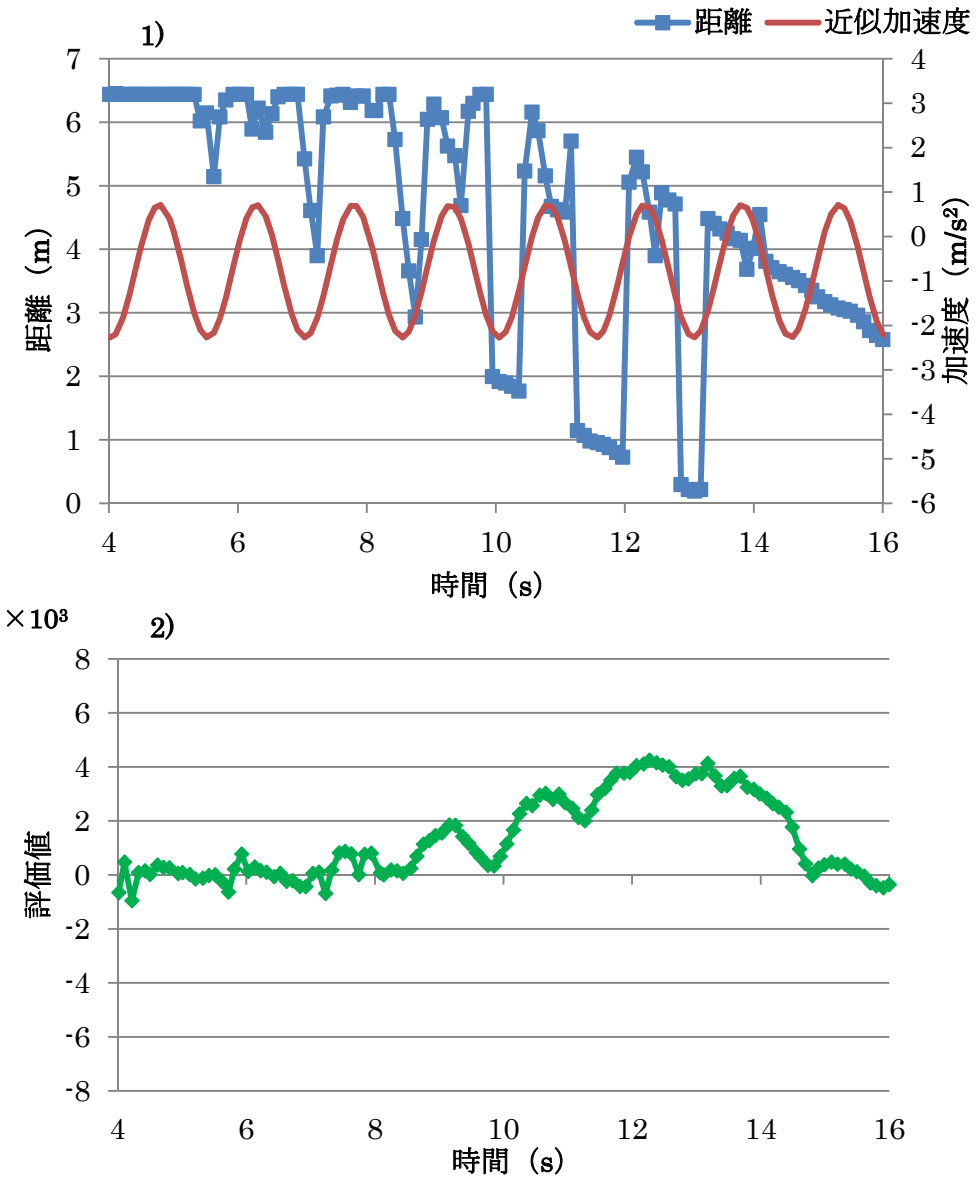


図 6.13　右方向に障害物　1)4-16s 間の詳細　2)評価値計算結果

以上の 2 パターンの実験結果より，加速度を用いて白杖の動きを認識し，その時の距離データとの相関から左右の障害物方向を特定することが出来る．

第7章　白杖のカスタマイズについて

7.1　カスタマイズとは

これまで述べたように，白杖に超音波センサと加速度センサを搭載し，評価値計算を行うことにより障害物の方向を検知できる．これは人間による視覚障害者支援と同様の有益な情報を提供できると考える．我々はこの障害物方向を検知する機能に加え，さらにセンサ類の装着がない白杖では検知できない範囲の障害物を検知する機能の追加を目指す．

そこで考えたのが，白杖のカスタマイズである．白杖の使用方法に個人差があるように，視覚障害者の方にとって有益な情報にも個人差がある．一例をあげると，視覚障害者が左右の壁から 2m ほど離れた場所を歩きたい際に，左右の検知が出来ると便利などの意見がある [13]．しかし，1.1.4 でも述べた通り，有益ではない細かすぎる情報は視覚障害者を混乱させる恐れがある．そこで，視覚障害者の方が，それぞれの白杖の使用方法に合わせた検知方向の選択を行えるように白杖にカスタマイズ機能を追加する．

カスタマイズとして前方の障害物検知及び障害物の前方左右方向の特定を標準検知範囲とし，胸部や頭部の高さ辺りの前方上部，視覚障害者の左右側面の 3 方向における障害物検知をオプションとして選択することができるようにする．それぞれの詳細については次節で示す．

なお，この 3 方向においては評価値による障害物方向を認識せず，距離のみによる障害物の有無判別とする．

7.1.1　上部障害物検知

前方上部障害物とは視覚障害者の胸部や頭部の高さに存在する障害物である．白杖の検知範囲は腰より下における前方約 1.5m と白杖が実際に届く範囲のみとなってしまう．図 7.1 に示す階段など下部が空いている場合やトラックの荷台からはみ出している荷物やお店の看板など，この高さにある障害物は白杖の死角になってしまい，検知することができない．そのため，視覚障害者が認識できずにぶつかることが多いと 1.1.4 でインタビューを行った歩行訓練士の方からご意見を頂いた．

また，前方上部の障害物検知に関しては，秋田精工株式会社製の“スマート電子白杖”により既に製品化されている（1.1.3 参照）．前方上部の障害物検知に関しては視覚障害者の方々の要望も多いため，本研究においても前方上部の障害物検知を行う．

図 7.1　白杖の死角となる階段（工学院大学犬目第 2 校舎 1 階）

7.1.2　左右側面の距離検知

左右側面の距離とは視覚障害者の側面に存在する物体までの距離である．視覚障害者にとって壁などは自己位置を認識するために非常に重要な役割を果たす．例えば，視覚障害者が壁沿いなどを歩く際には，壁と地面を白杖で交互に叩きながら沿い歩きを行うことにより方向や現在位置の判断が容易にできる．また，塀の角や道路の交差点などは視覚障害者にとって単独歩行を行う際に重要な目印となる．しかし，商店街など沿い歩きができない場所もある．その場合，視覚障害者は白杖のみでの単独歩行を強いられため，左右の壁から 2m ほど離れた場所を歩きたい時に，左右側面の検知が出来ると便利との意見がある[13].

しかし，側面の検出に関しては歩行訓練士の方から，周囲の音や通行人の話し声などから現在地やその場の状況を認識できるため，不要な情報になるとのご指摘を受けた．左右側面の距離検出は視覚障害者の生活環境に大きく影響を受けるため，有益な情報であるかは各個人により異なる．そこで，カスタマイズにより使用者が選択できるようにする．

7.2　システムの変更点

図 7.2 に変更後のシステム全体構成を示す. 2.1 で示したシステムに前方上部の障害物と左右側面の距離を検出するために超音波センサを 3 つ追加する．しかし，これまでのシステムのまま超音波センサを同時に複数個使用すると相互干渉を起こしてしまう．相互干渉とはあるセンサが発信した超音波を，別のセンサが受信し存在しない物体を存在すると誤検知をすることである．解決方法としてセンサを互いに接続し，4 つのセンサのうち同時刻に動作しているセンサが 1 つだけになるよう同期させる [18]. 付録 2 に回路図を添付する．

また，実際に白杖へ設置を行ったものが図 7.3 となる（付録 3 に組立図を添付）．7.1 で示した 4 方向の障害物を検知するため，それぞれの方向へ超音波センサを設置している．

次節で 4 つの超音波センサの同期をさせるためのファームウェア，Java アプリケーションの変更点を述べる．

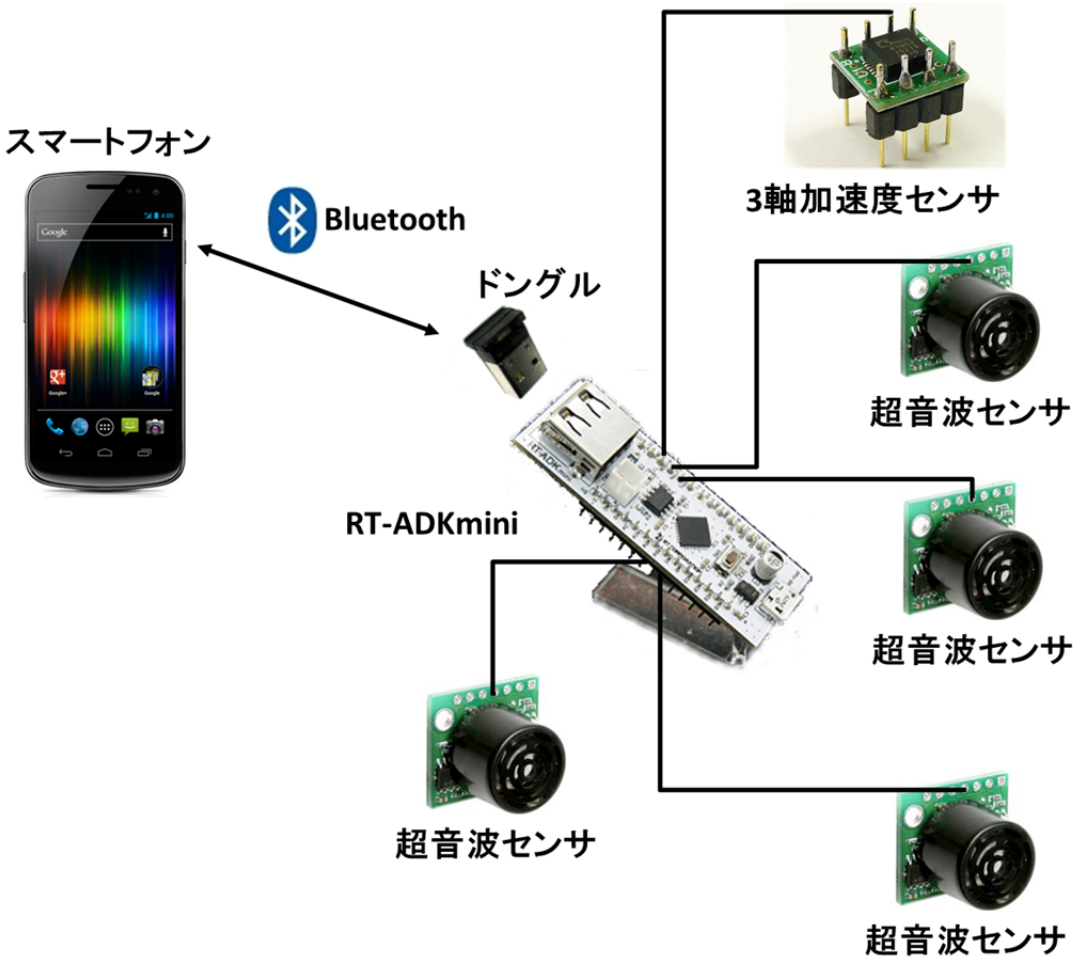


図 7.2　システムの全体構成（変更後）

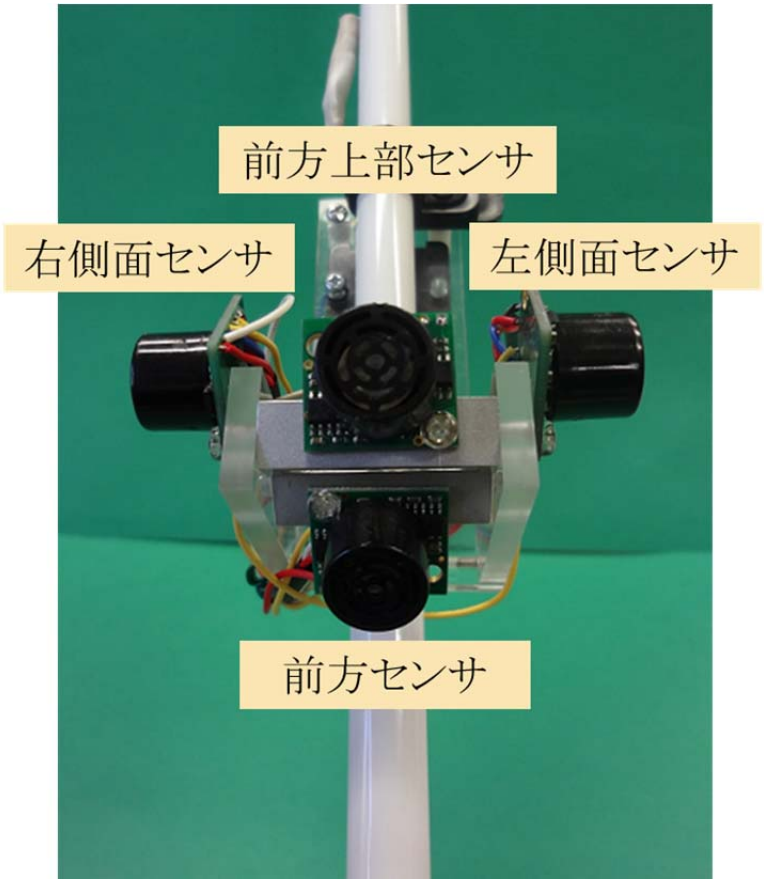


図 7.3　白杖への設置（超音波センサ 4 つ版）

7.3　ファームウェア変更点

7.3.1　周期パルスによるセンサの始動

超音波センサの同期方法は文献 [27]に記されている“Sequentially Read Each MaxSonar”法を使用する．図 7.4 に同期線の配線図を示す．まず，最初のセンサの RX ピンにパルス波を入力する．最初のセンサの距離取得が終了すると TX ピンから次のセンサの RX ピンに順次パルス波が送信され，最後のセンサまで順番に超音波センサを動作させることができる．入力波には最後のセンサが距離を取得し終わるまでの待ち時間を作らなければならない．使用センサ（LV-MaxSonar-EZ1）は一回の距離取得動作時間が 50ms であるため，その 4 倍の 200ms 以上の待ち時間が必要となる（図 7.5） [27].

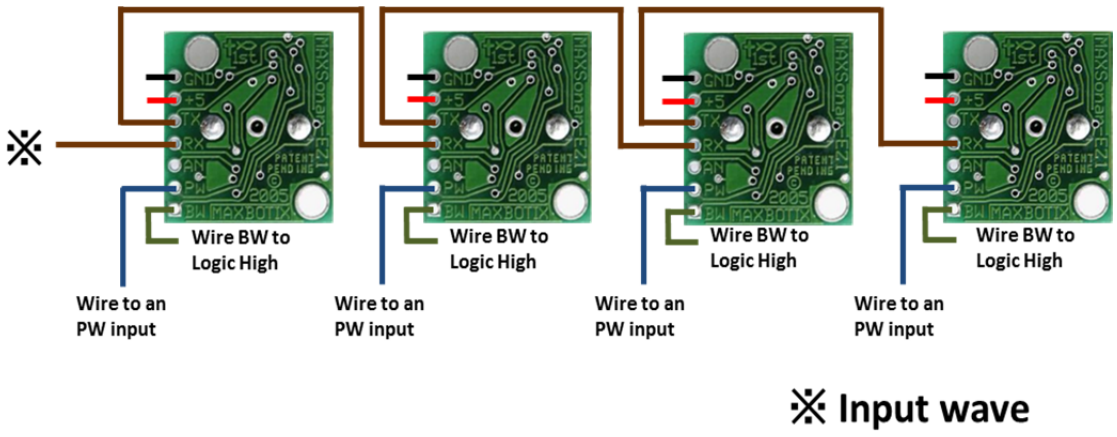


図 7.4　同期線の配線図 [27]

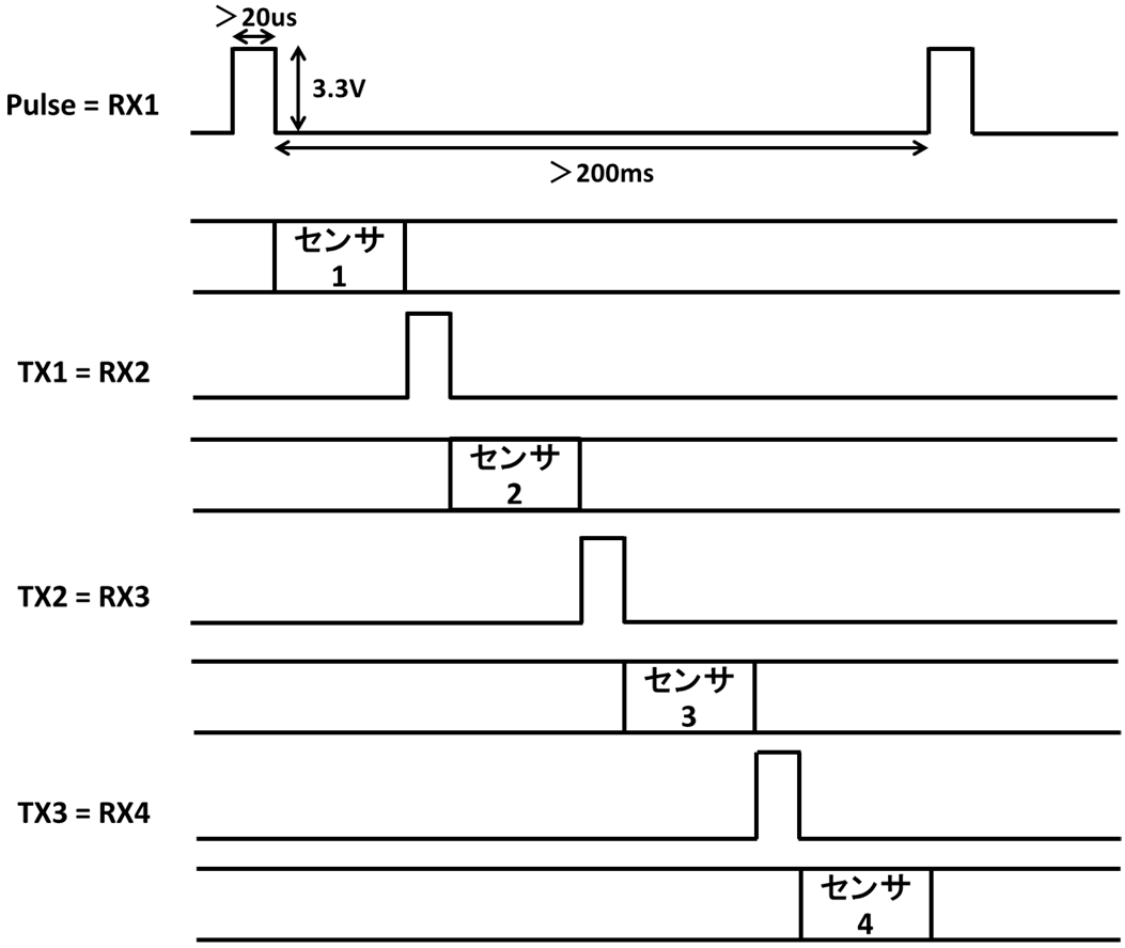


図 7.5　周期パルスによる超音波センサの動作順番

そこで，最初のセンサへの入力としては周期パルスを与える．周期パルスの生成には，時間によって出力ピンのオンオフを制御できる PIC 内蔵の出力コンペアモジュール（OC1）を用いる [22].

周期パルスの設定は以下となり，パルス幅 30 μ s，周期 200ms，振幅 3.3V の入力波形を作成することができる．

```
PPSOutput(PPS_RP4, PPS_OC1);//4/20 PWM出力設定:RP4-OC1

//PWM Timer
OpenTimer4(T4_ON | T4_PS_1_64 ,0xffff); //

//OC1, Config+
OpenOC1(OC_IDLE_CON | OC_TIMER4_SRC | OC_PWM_EDGE_ALIGN ,OC_SYNC_TRIG_IN_TMR4,0,0);
    //PWM Base Freqencey
PR4=0xA584;//PWM_Config
SetDCOC1PWM(0xA,0); //幅の設定
```

7.3.2 Input Capture によるパルス幅の読み取り

超音波センサにはデジタル出力とアナログ出力があるが，第 3 章では便宜上アナログ出力を用い，加速度センサのデータと同等に扱ってきた．しかし，4 つのセンサを周期パルスにより同期させてデータを取得した際，アナログ出力では安定した距離データを取得できなかった．同期方法の変更などを行いアナログ出力による安定したデータ取得を目指したが実現できず，原因が超音波センサ側のアナログ出力にあると判断した．そこで，以下ではアナログ出力を用いずデジタル出力（PWM[12]）を用いることにする．

これまで用いてきた A/D 変換は加速度センサのみで使用し，超音波センサの出力するパルス信号の幅計測にはインプットキャプチャを使用する．インプットキャプチャを利用すると，入力されるパルスのエッジ間の時間を計測することができるため，パルス幅やパルス周期の測定を行うことができる．入力ピンに入力されるパルスの立ち上がりか立下りのエッジでタイマカウンタの値をバッファにコピーし，このタイマを一定周期で動作させることによりパルスのエッジ間の時間差を計測することができる．PIC 内にはインプットキャプチャモジュールが 5 個内蔵されているため，こちらを使用し 4 つの超音波センサのパルス幅を読み取る [22] [23].

プログラムの一部を参照しながら詳細を示す．インプットキャプチャには IC1~IC4 までを用い，毎回のエッジごとでのタイマ時間の取得を行う．

まず使用する変数を宣言する．立ち上がりと立ち下がりの両方のエッジを検出する場合，インプットキャプチャモジュールの仕様により 2 つのエッジを区別することはできない．そこで，エッジが奇数回目か偶数回目かを判別するために初期値は-1 としている．

```
signed long Fst1 = -1;
signed long Snd1 = -1;
signed long Fst2 = -1;
signed long Snd2 = -1;
signed long Fst3 = -1;
signed long Snd3 = -1;
signed long Fst4 = -1;
signed long Snd4 = -1;
```

次に，インプットキャプチャの設定を ICxCON レジスタで行う．毎回の立ち上がり，立ち下がりごとのタイマ時間の取得とし，ここで入力ピンの割り付け設定も行う．

[12] Pulse Width Modulation の略．変調方法の一つであり，パルス波のデューティー比を変化させて変調する．

```
//Input Config1
 CLKDIV =0;

 T2CON = 0x8020;
 PR2 = 0xFFFF;
 TMR2 = 0;

 RPINR7bits.IC1R = 1;

 IC1CON1 = 0x2421;
 IC1CON2 = 0x0;
 IEC0bits.IC1IE =1;

//Input Config2

 RPINR7bits.IC2R = 2;

 IC2CON1 = 0x2421;
 IC2CON2 = 0x0;
 IEC0bits.IC2IE =1;

//Input Config3

 RPINR8bits.IC3R = 3;

 IC3CON1 = 0x2421;
 IC3CON2 = 0x0;
 IEC2bits.IC3IE =1;

//Input Config4

 RPINR8bits.IC4R = 15;

 IC4CON1 = 0x2421;
 IC4CON2 = 0x0;
 IEC2bits.IC4IE =1;
```

取得したタイマの値からパルス幅の計算を行うために，何回目のエッジかの判別を行う．2 回目のエッジを検出した際に 1 回目のタイマ時間との差を計算し，その値がパルス幅とする．この時に，入力キャプチャのイベントが起きるタイミングと，タイマのカウントのタイミングは無関係なので，タイマカウンタがロールオーバーして前後のキャプチャ値の大きさが逆転していないかを確認している．以上の結果を変数 us_data_x に格納し，スマートフォンに送信している．送信処理は 3.3 と同様である．

```
void __attribute__((interrupt, no_auto_psv)) _IC1Interrupt(void){

 if(IC1CON1bits.ICBNE){
    if(Fst1<0){                                  //一回目のエッジ検出時
        Fst1 = IC1BUF;
        IFS0bits.IC1IF = 0;
        return ;
    }else{
        Snd1 = IC1BUF;                           //二回目のエッジ検出時
        IFS0bits.IC1IF = 0;
         if(Fst1 <= Snd1)
             us_data = ((Snd1 - Fst1)*511)/9505;         //6.5mの値が9505
         else
            us_data = (((0xFFFF -Fst1) + Snd1)*511)/9505;    //ロールオーバしている場合

        Fst1 = Snd1 = -1;
    }
  }
}
void __attribute__((interrupt, no_auto_psv)) _IC2Interrupt(void){
     if(IC2CON1bits.ICBNE){
    if(Fst2<0){
        Fst2 = IC2BUF;
        IFS0bits.IC2IF = 0;
        return ;
    }else{
        Snd2 = IC2BUF;
        IFS0bits.IC2IF = 0;
         if(Fst2 <= Snd2)
             us_data_up = ((Snd2 - Fst2)*511)/9505;
         else
            us_data_up = (((0xFFFF -Fst2) + Snd2)*511)/9505;

        Fst2 = Snd2 = -1;
    }

  }
}
void __attribute__((interrupt, no_auto_psv)) _IC3Interrupt(void){
     if(IC3CON1bits.ICBNE){
    if(Fst3<0){
        Fst3 = IC3BUF;
        IFS2bits.IC3IF = 0;
        return ;
    }else{
        Snd3 = IC3BUF;
        IFS2bits.IC3IF = 0;
         if(Fst3 <= Snd3)
             us_data_right = ((Snd3 - Fst3)*511)/9505;
         else
            us_data_right = (((0xFFFF -Fst3) + Snd3)*511)/9505;

        Fst3 = Snd3 = -1;
    }

  }
}
```

```
void __attribute__((interrupt, no_auto_psv)) _IC4Interrupt(void){
     if(IC4CON1bits.ICBNE){
    if(Fst4<0){
        Fst4 = IC4BUF;
        IFS2bits.IC4IF = 0;
        return ;
    }else{
        Snd4 = IC4BUF;
        IFS2bits.IC4IF = 0;
         if(Fst4 <= Snd4)
             us_data_left = ((Snd4 - Fst4)*511)/9505;
         else
            us_data_left = (((0xFFFF -Fst4) + Snd4)*511)/9505;

        Fst4 = Snd4 = -1;
    }

  }
}
```

7.4 Android アプリケーション変更点

7.4.1 フローチャートの変更点

データ処理部変更後フローチャートを図 7.6 に示す．まず，7.4.2 に示す距離ジャンプ処理を行った後，前方方向の障害物検知用の超音波センサの判別を行い（処理 A），次に前方上部の障害物判別を行う流れとなる．4.1 との相違点は，評価値のみでは状況判断を行うことができなかった正面に障害物が存在する場合に備え，超音波センサの取得距離だけによる判別を追加した．これにより，使用者が設定した設定値 1（m）を超える値を検知した際，まず前方に障害物が存在すると判断する．その後，評価値による判別を行い，障害物方向が特定できればさらに通知を行う．

また，使用者への通知方法として，音声と振動を組み合わせ，伝達を二重に行うことにより，使用者が情報を取り損ねないようにする．また，実験中の確認や展示会などで健常者に説明を行うため，スマートフォンのディスプレイにも判別結果の表示を行う．

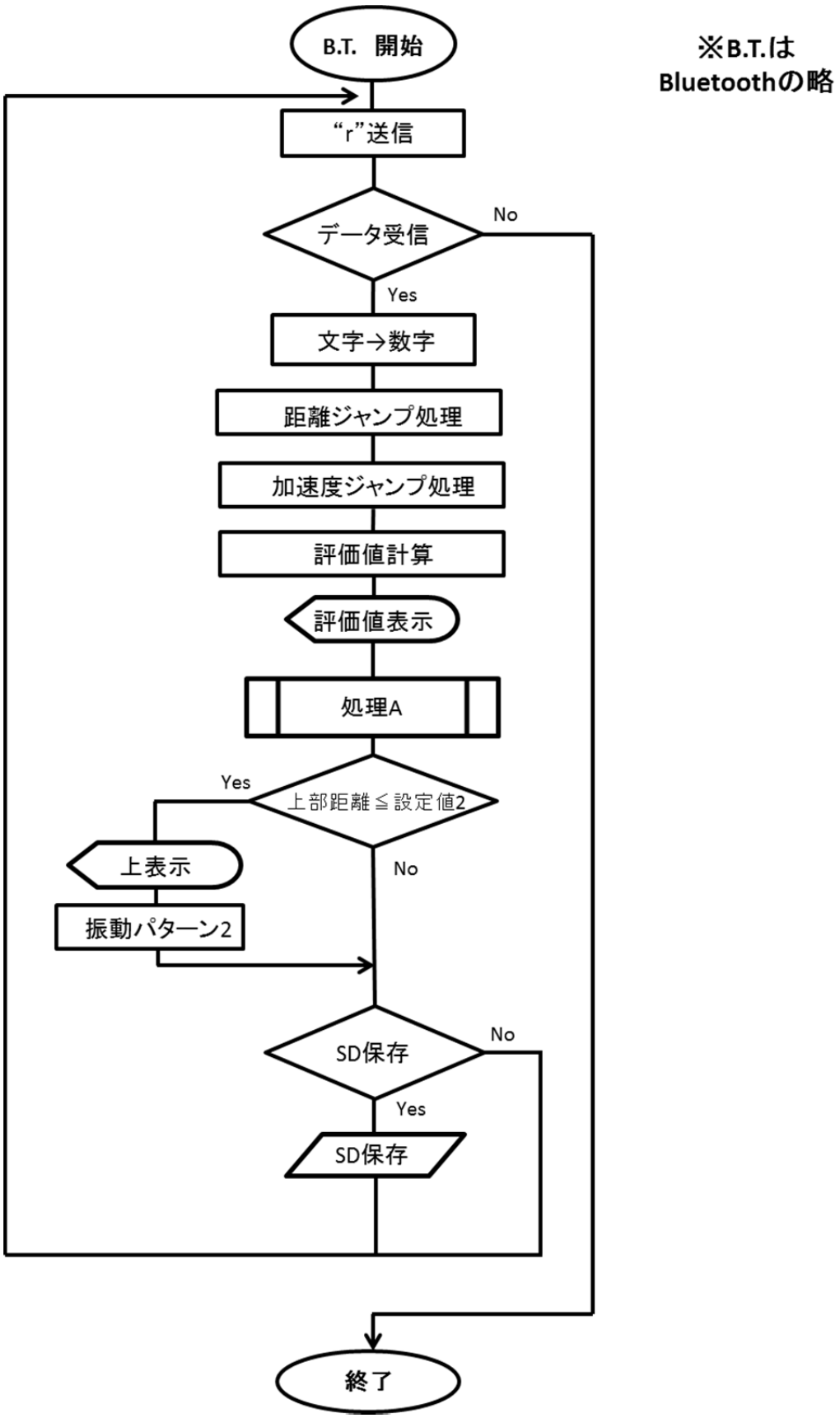


図 7.6　データ処理部フローチャート変更版（前方，前方上部の障害物のみ通知）

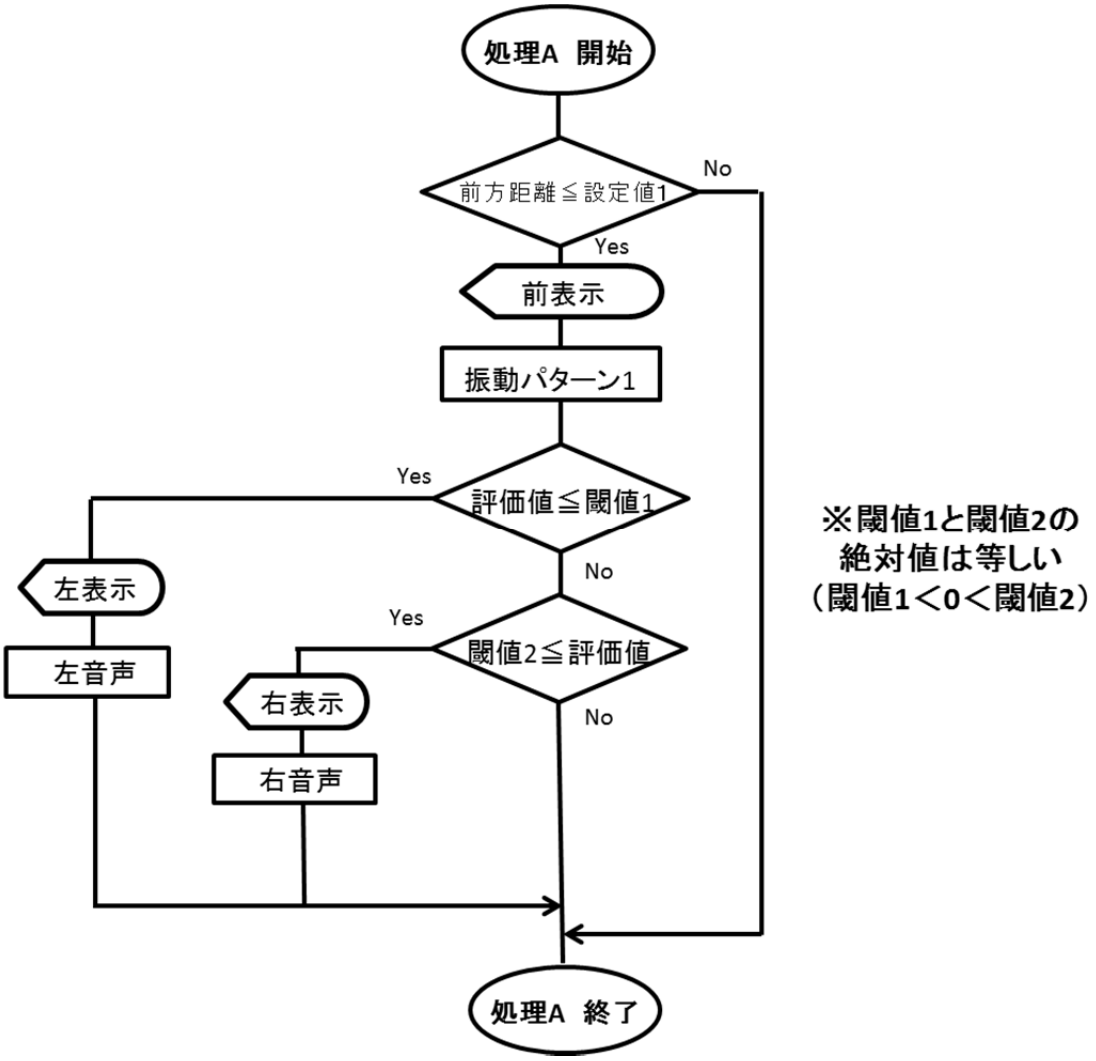


図 7.7　前方検知（前方左右を含む）の詳細（処理Ａ）

7.4.2　超音波センサのジャンプ処理

超音波センサを同時に複数個使用するため周期パルス信号を用いて同期を行ったが，意図しない取得距離を受信してしまう時があった（図 7.8）．そのため，4.2 同様のジャンプ処理をすべての取得距離に対して行う．距離データは一つ前のデータとの比較だけでは正誤の判別ができないため，過去二つ分のデータと比較を行う．

```
//前方センサジャンプ処理
float prevsonic = obtainedValue4;
float sonic_data = prevsonic;
int THRD_sonic = 100;

if(Math.abs(prevsonic-old_data1)>THRD_sonic &&
        Math.abs(old_data1-old_data2)<THRD_sonic ){
    sonic_data = old_data1;
}

old_data2 = old_data1;
old_data1 = prevsonic;

//上部センサジャンプ処理
float prevsonic_up = obtainedValue5;
float sonic_data_up = prevsonic_up;
if(Math.abs(prevsonic_up-old_data1_up)>THRD_sonic &&
        Math.abs(old_data1_up-old_data2_up)<THRD_sonic ){
    sonic_data_up = old_data1_up;
}

old_data2_up = old_data1_up;
old_data1_up = prevsonic_up;

//右センサジャンプ処理
float prevsonic_left = obtainedValue6;
float sonic_data_left = prevsonic_left;
if(Math.abs(prevsonic_left-old_data1_left)>THRD_sonic &&
        Math.abs(old_data1_left-old_data2_left)<THRD_sonic ){
    sonic_data_left = old_data1_left;
}

old_data2_left = old_data1_left;
old_data1_left = prevsonic_left;
//左センサジャンプ処理
float prevsonic_right = obtainedValue7;
float sonic_data_right = prevsonic_right;
if(Math.abs(prevsonic_right-old_data1_right)>THRD_sonic &&
        Math.abs(old_data1_right-old_data2_right)<THRD_sonic ){
    sonic_data_right = old_data1_right;
}

old_data2_right = old_data1_right;
old_data1_right = prevsonic_right;
```

ジャンプ処理では，prevsonic に現在の取得距離値，old_data1 に一つ前の取得距離値，old_data2 に二つ前の取得距離値を格納し，任意の閾値 THRD_sonic で設定した以上になった場合には old_data1 に置き換えを行う．

このジャンプ処理を行うことにより，図 7.9 のように意図しない取得距離の除去が行える．

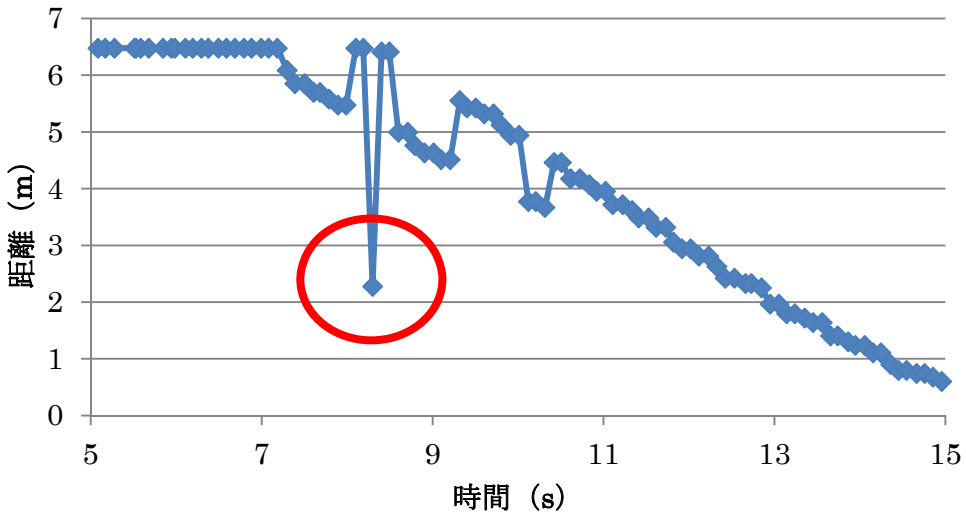


図 7.8　取得距離の誤検知

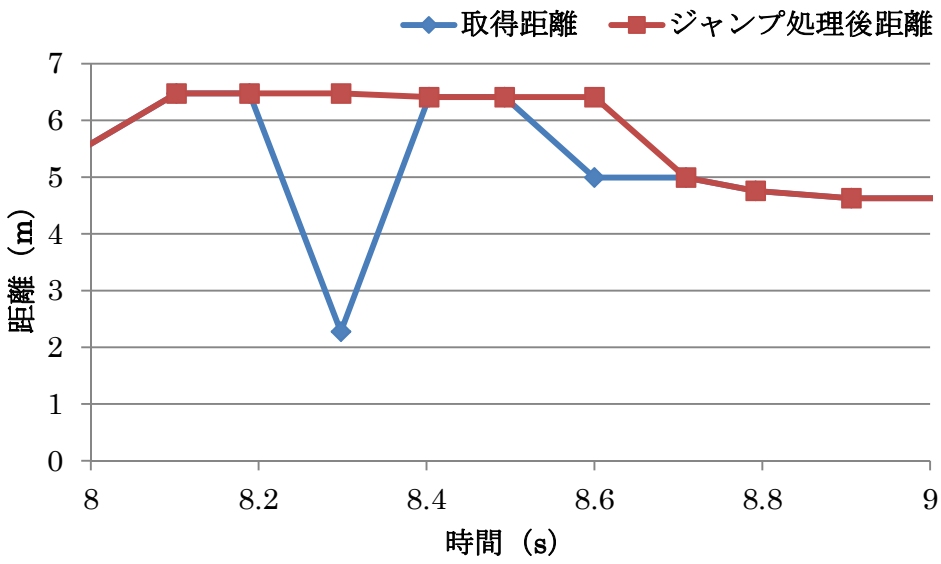


図 7.9　8-9s 間の取得距離とジャンプ処理後距離の比較

7.5　通知方法

使用者への通知方法として，スマートフォン内蔵の音声出力と振動子の両方を用いる．振動のみによる伝達を考えていたが，1.1.4 のインタビューより，認識できるのは 3 パターンまでとの意見を頂いた．一方，音声のみによる伝達では街中など騒音の多い場所では使用者が聞き取れないことなどが考えられる．そこで，音声と振動を二重に用いることにより，使用者が情報を取り損ねないようにする．また，実験中の確認や展示会などで健常者に説明を行うため，スマートフォンのディスプレイにも判別結果の表示を行う．本節でそれぞれの詳細を示す．

7.5.1　音声と振動

超音波センサが障害物を検知及び評価値による障害物方向が特定できた場合，スマートフォンに内蔵されている振動子と音声を併用することにより視覚障害者への警告を行う．警告には，図 7.6 に示した順番で判別を行う．

まず，音声出力による警告方法を示す．まず，再生したい音楽ファイルを raw ディレクトリ以下に保存し（図 7.10），MediaPlayer クラス[13]が提供するメソッドを利用することにより再生を行うことが出来る [28]．

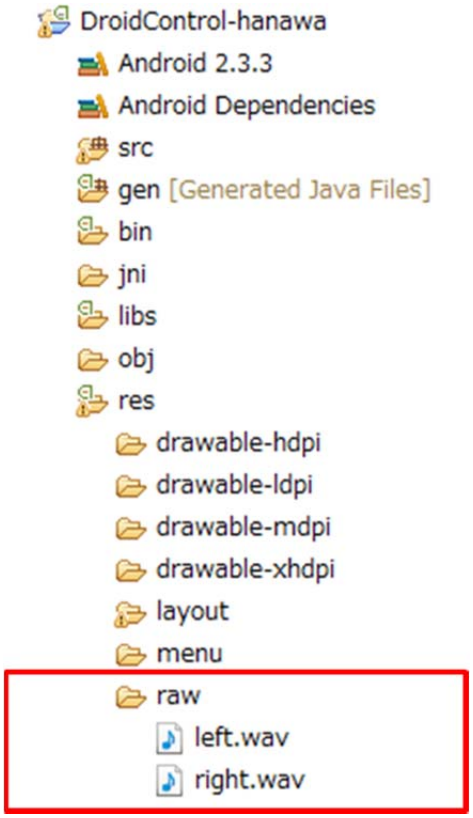


図 7.10　音声データ保管場所

[13] Android.media パッケージを import することで利用できる．

次に，振動による伝達方法について詳細を示す．スマートフォン内部の振動子を使用するためには以下の記述を行うことにより使用できる．振動パターンも自分で決定できるため，今回は振動パターン 1 と振動パターン 2 の 2 パターンを作成した．

```
long[] pattern = {50, 1000,};       //振動部分
vib.vibrate(pattern, -1);           //繰り返さないための処理
```

以上の準備により音声と振動を併用することが出来る．そこで，この二つの使い分けについて図 7.11，図 7.12 に示す． 振動による伝達では，赤枠で囲んだ前方の障害物を検知した場合に振動パターン 1 を実行し，前方上部に障害物を検知した場合には，振動パターン 2 を実行させる．この 2 パターンに分類することにより，視覚障害者が直観的に障害物の場所を認識できると考える．また，左前方障害物と右前方障害物の方向を特定できた場合には，警告音による警告を行う．左側面，右側面に関しては，7.1.2 で述べた通り各使用者により情報の必要性が変化するため，今回は未実装とした．

音声と振動を併用することにより，障害物検知時の伝達ミスを防げると考える．

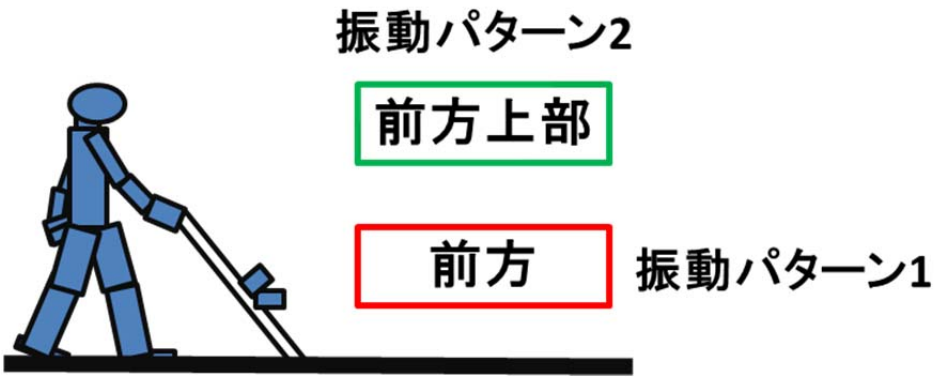


図 7.11　前方上部方向の伝達方法

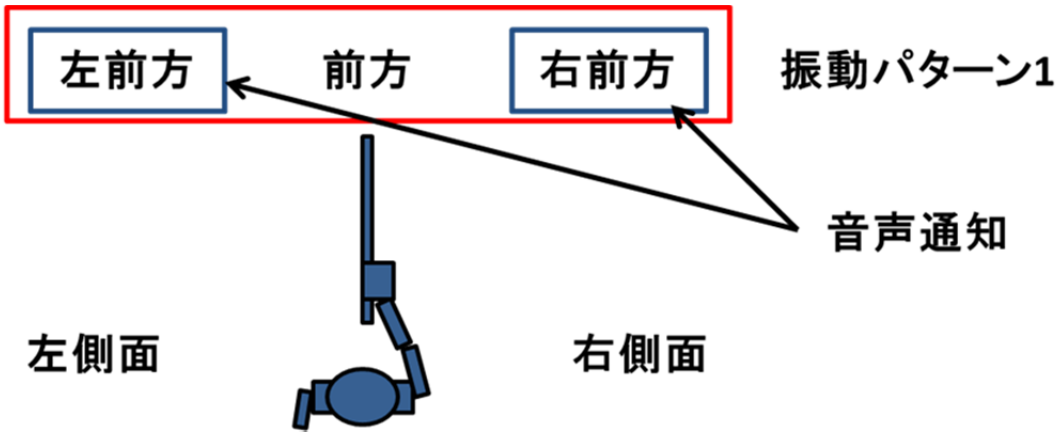


図 7.12　前方方向の伝達方法の分別（左右側面未実装）

7.5.2　画面表示

実験中の確認や展示会などで健常者に説明を行うため，スマートフォンのディスプレイにも判別結果及び評価値の表示を行う．図 7.13 に Java アプリケーションの UI を示す．①にチェックを入れると内部メモリへの保存が開始される（4.4 参照）．②にはデータ取得毎に行われる評価値計算結果が表示される．③には使用者が任意に設定できる前方方向の検知距離設定結果が表示され，現在は，2.5m 先の障害物までを検知している．

図 7.13　Java アプリケーション UI の詳細（障害物なしの場合）

次に，それぞれの検知方向に障害物を認識した場合について示す．図 7.14 1)は白杖から前方 2.5m 以内に障害物を検知した場合に表示される．図 7.14 2)は前方上部の障害物を検知時に表示される．この 2 方向を同時に検知した場合が図 7.15 である．図 7.16 は前方の障害物を検知し，なおかつ障害物方向を特定できた場合の表示である．左右前方の検知と同時に上部の障害物を検知した物が図 7.17 となる．これらは，7.4.1 で述べた判別方法によって分類されている．

1)　前方に障害物

2)　前方上部に障害物

図 7.14　障害物検知時のディスプレイ表示内容

図 7.15　前方-前方上部障害物の同時検知

1)　左前方に障害物

2)　右前方に障害物

図 7.16　左右前方に障害物検知時のディスプレイ表示内容

1)　左前方上部-左前方に障害物

2)　左前方上部-左前方に障害物

図 7.17　上部-左右前方に障害物検知時のディスプレイ表示

第8章　カスタマイズ評価実験

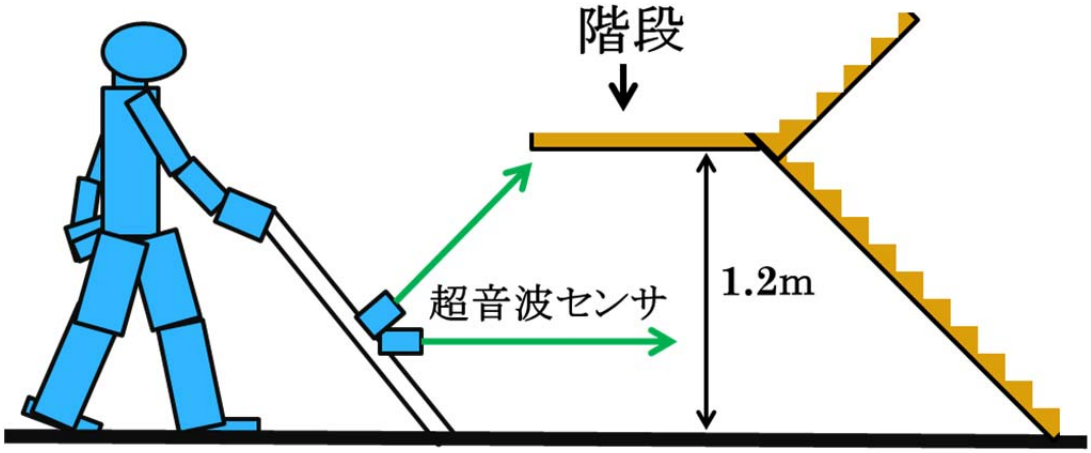
これまで述べた白杖のカスタマイズについて各計測データ取得状況の確認実験を行う．特に，カスタマイズを行った前方上部と左右側面に装着したセンサの取得データ確認を目指す．

8.1　前方上部方向の確認実験

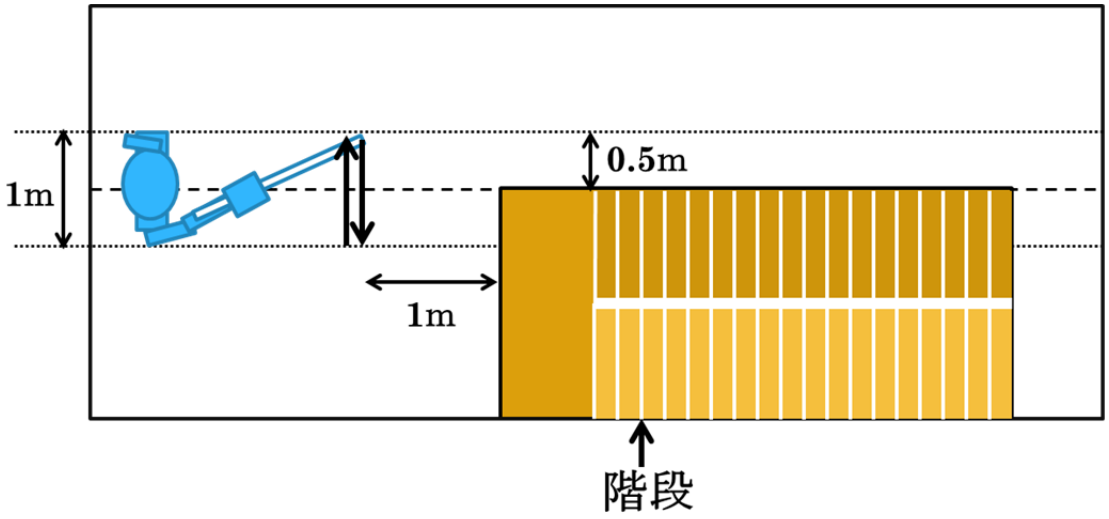
本節では図 8.1 に示す前方上部方向の障害物検知を目指す．実験場所には図 8.2 1)に示す下方が空いている階段から約 1m 離れた位置を選び，その場で白杖を左右に振っている．このような場合，前方上部の障害物は前方方向の超音波センサの検知範囲外となってしまい検知することができず，障害物なしと判断してしまう．

白杖を振る範囲は，前方上部にある階段の障害物と障害物が存在しない所がどちらも検知できる範囲とする（図 8.2 2)）．超音波センサの白杖への取り付け位置は地面から 0.5m の位置とする．

図 8.1　検知方向詳細（工学院大学犬目第２校舎１階）



1)　正面図



2)　平面図

図 8.2　実験環境詳細

8.2　実験結果及び考察

図 8.3 に実験結果を示す．データ取得開始 3s 後から 20s まで白杖を振っている．前方上部方向の距離は 1m 付近と 3.5m 付近を往復していることが確認できる．これは白杖が右方向に振られ，階段を検知することにより取得距離が短くなり，左に振られることで障害物なしの状態となっていることを示している．この際，前方方向距離では 3m～4m の間にほぼ収まっていることが確認できる．これは，前方方向の超音波センサではこの高さの突出物を検知することができないことを示している．

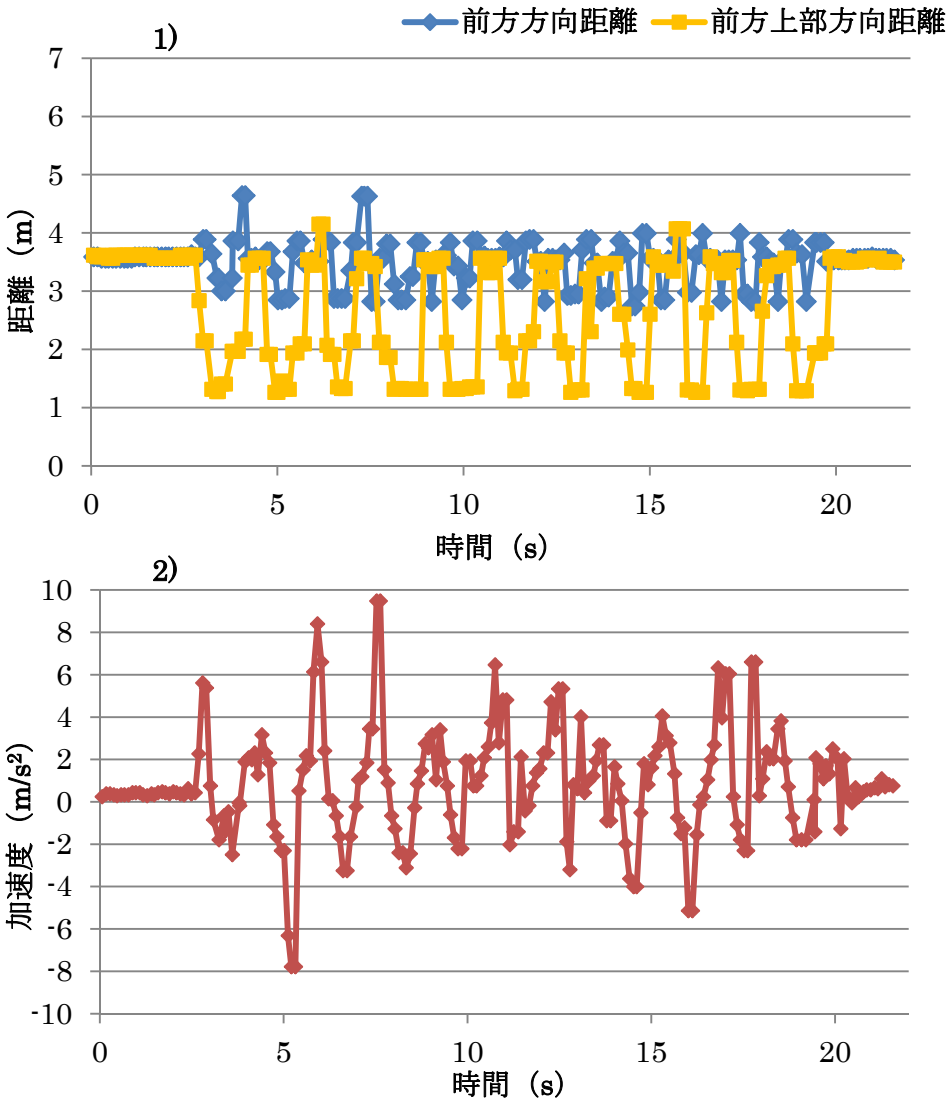


図 8.3 1）前方方向距離-前方上部方向距離グラフ 2）加速度グラフ

図 8.3 の 5-15s 間を拡大したものが図 8.4 となる．加速度がマイナスになっている時に前方上部方向距離が短くなっていることが確認できる．これらより，白杖の動きに連動して取得距離が変動していることが分かる．

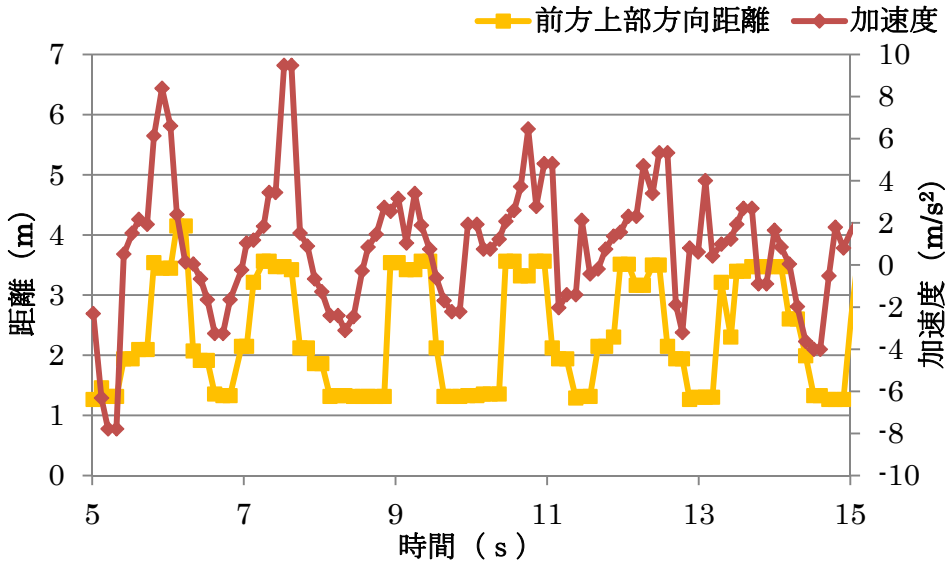


図 8.4　5-15s 間の前方上部方向距離-加速度グラフ詳細

以上より，前方上部方向の距離を計測することにより，前方方向の超音波センサでは検知できない高さの障害物を，検知することができる．

8.3　左右側面方向の取得距離確認実験

本節では図 8.5 に示す視覚障害者の左右側面方向の距離検知を目指す．視覚障害者は白杖のみでの単独歩行を強いられため，左右の壁から 2m ほど離れた場所を歩きたい時に，左右側面の検知が出来ると便利との意見がある．しかし，左右側面方向に関しては，7.1.2 で述べた通り視覚障害者によっては必要が無いという意見もある．そこで，左右側面方向に関しては取得データの取得のみとし，本研究においても視覚障害者への伝達は行わない．

実験環境を図 8.6 に示す．白杖を振りながら幅が約 1.8m の廊下の中心を歩く．その際，白杖は左右 0.3m の範囲で振り，視覚障害者の左右側面は壁となっている．

図 8.5　左右側面方向の詳細

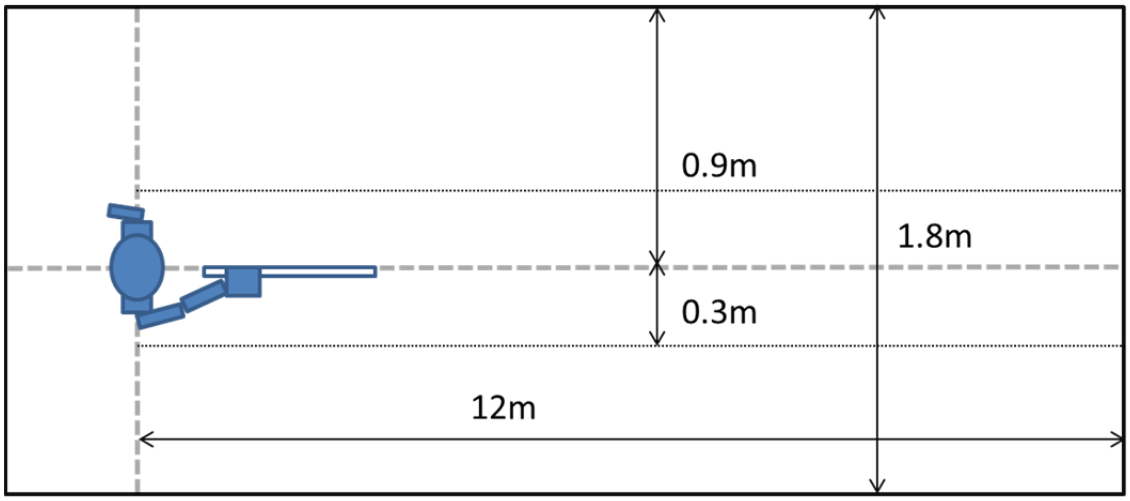


図 8.6　実験環境詳細

8.4　実験結果及び考察

図 8.7 は左右側面の壁までの取得距離グラフである．白杖を振りながら壁から約 1m の位置を歩いているため，3-14s 間では取得距離に変動がみられる．これは白杖の振り幅による変動であり，左右側面までの取得距離の増加と減少が重なっていることが確認できる．

以上より，左右側面の距離検出を可能とした．左右側面方向の距離取得は，非接触による沿い歩きができるため，商店街など白杖で突けない場所においては有用である．

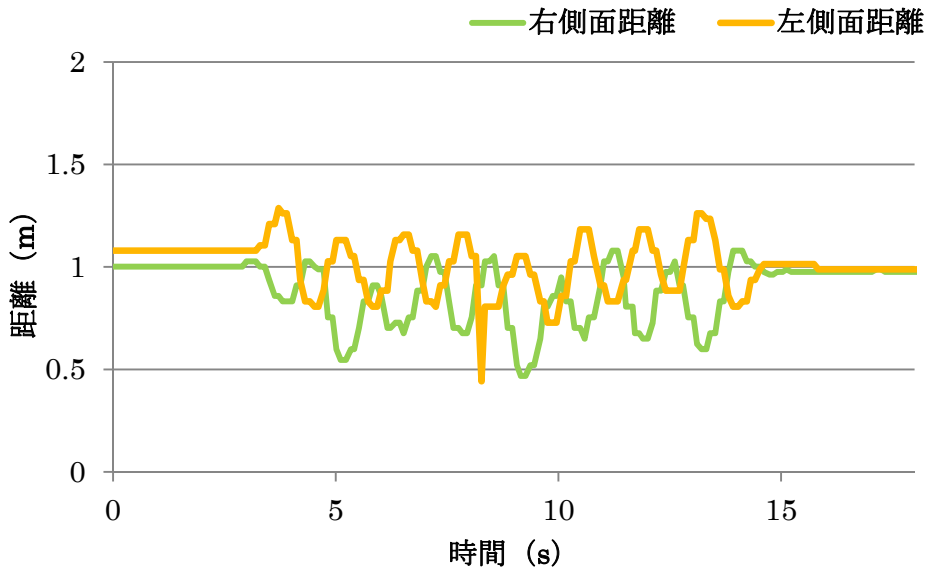


図 8.7　左右側面までの検知距離

第9章　総括

近年白杖などの視覚障害者用補助機器を電子化する試みに注目が高まっている．白杖に関してはセンサやカメラなどを使用した研究が行われ始めているが，データ処理に PC を用いたりするため実用化に至る物は少ない．また，視覚障害者にとって電子補助機器からの有益な周辺情報の提供が十分ではなく，まだまだ周囲の人々の援助に頼ることが多いのが現状である．これは取得情報（データ）のみによる情報提供では具体性に欠けるからであると考えた．

そのため，本研究ではスマートフォン（Android OS を搭載）を用いてデータ処理部を小型化し，視覚障害者に障害物方向及び距離の有益な情報を提供できるシステム構築を目指した．解決手法として，障害物方向を特定するために，白杖に外部の超音波センサと加速度センサを搭載し，障害物までの距離と白杖の動きを取得した．また，そのデータを自由度の高い無線通信を用いてデータ処理部に送信し，評価値により障害物方向の特定を行った．

また，我々はこの前方の障害物検知及び障害物方向の特定を標準検知範囲とし，胸部や頭部の高さ辺りの前方上部，視覚障害者の左右側面の 3 方向における障害物検知をオプションとして選択できるようにした．

本章では，前方方向，前方上部方向の障害物検知，左右側面方向の距離検知に関してそれぞれ考察を行う．また，製品化までの課題を述べ，総括とする．

9.1　前方方向の障害物及び障害物方向の検知

前方方向の障害物方向を特定するために取得距離による判別と評価値による判別を併用した．評価値とは加速度データと超音波センサからの距離データを用いて計算を行うもので，その結果を基に障害物方向を判定した．評価値を求める流れとして，「sin 波への近似（FFT による周波数決定→Newton 法による調整）→評価値計算」となる．評価値計算は sin 波への近似を行った Y 軸方向の加速度値と超音波センサの距離の積を積分することにより算出した．この評価値を 0 付近に収まる「障害物なしの場合」，評価値がプラスとなる「右前方方向に障害物が存在する場合」，評価値がマイナスとなる「左前方方向に障害物が存在する場合」の 3 パターンに任意の閾値で判別することにより障害物方向の特定を可能とした．

評価値は本システムの使用環境により大きく変動するため，閾値の決定方法に関しては任意で決めた．評価値は距離データと加速度データの積を積分し求めるが，障害物方向までの取得距離と障害物が存在しない方向の取得距離の差が大きいほど評価値は大きくなるため，広い道や人の少ない所では評価値が大きくなる傾向であった．しかし，人混みなどのこの距離の差が大きく出にくい所では評価値が小さくなる傾向となった．よって，使用者の方がどのような環境で主に使用されるかによって閾値が変わるため，本研究ではその場に応じた適切な閾値を設けることができるように，作成したアプリケーション内で閾値

の変更を可能とした． しかし，これでは視覚障害者の方が使用できないため，閾値決定の方法の改善が必要である．

9.2 前方上部方向の障害物検知

我々は前方方向の障害物方向を検知する機能に加え，さらにセンサ類の装着がない白杖では検知できない範囲の障害物を検知する機能の追加（カスタマイズ）を目指した．前方上部障害物とは視覚障害者の胸部や頭部の高さに存在する障害物である．白杖の検知範囲は腰より下における前方約 1.5m と白杖が実際に届く範囲のみとなってしまうため，階段など下部が空いている場合やトラックの荷台からはみ出している荷物やお店の看板など，この高さにある障害物は白杖の死角になってしまい，検知することができない．

そこで，前方の障害物検知及び障害物方向の特定を標準検知範囲とし，胸部や頭部の高さ辺りの前方上部方向の距離を計測することにより，センサ類の装着がない白杖では検知できない高さの障害物を，検知することを可能とした．なお，前方上部方向の障害物検知には評価値計算を行わず，距離の値のみによる閾値を設け判別を行った．

前方上部方向の確認実験より，前方方向の超音波センサでは前方上部の障害物を検知できないことを示した．そのため，前方上部方向の距離を計測することにより，前方方向の超音波センサでは検知できない高さの障害物を検知できることを確認した．

9.3 左右側面方向の距離検知

左右側面方向の距離とは視覚障害者の側面に存在する物体までの距離である．視覚障害者が壁沿いなどを歩く際には，壁と地面を白杖で交互に叩きながら沿い歩きを行うことにより方向や現在位置の判断が容易にできるなど，壁などは自己位置を認識するために非常に重要な役割を果たす．しかし，商店街など沿い歩きができない場所もあるため，左右の壁から 2m ほど離れた場所を歩きたい時に，左右側面の検知が出来ると便利との意見があった．

そこで，左右側面の距離を計測することにより，壁などからある一定の距離を非接触により認識することを可能とした．しかし，左右側面方向の距離検出は，視覚障害者にとって必要が無いとの意見もあった．そこで，本研究においては取得データの取得のみとし，視覚障害者への通知は行わなかった．左右側面方向の障害物までの距離をオプションとして選択することができるようにした．また，左右側面までの検知距離を視覚障害者にとって有益な情報として通知することが今後の課題である．

9.4　通知方法

使用者への通知方法として，スマートフォン内蔵の音声出力と振動子の両方を用いた．振動のみによる伝達では認識できるのは 3 パターンまでと少なく，音声のみによる伝達では街中など騒音の多い場所では使用者が聞き取れないなどが考えられるため，音声と振動を二重に用いることにより，使用者が情報を取り損ねないようにした．また，実験中の確認や展示会などで健常者に説明を行うため，スマートフォンのディスプレイにも判別結果の表示を行った．左右側面方向に関しては，使用者により情報の必要性が変化するため，本研究では未実装とした．

前方方向に障害物を検知した場合には，振動パターン 1 で通知し，前方上部方向に障害物を検知した場合には，振動パターン 2 でスマートフォンを振動させた．この時，左前方，右前方の障害物方向を特定できた場合には，音声により障害物方向の通知を行った．これにより，視覚障害者は二重に通知を受け取ることを可能とした．また，スマートフォンのディスプレイには評価値計算結果，前方方向の検知距離設定距離，前方上部，前方の障害物及び前方左右の障害物方向検知結果など音声や振動と同じ判別結果を表示させた．

以上より，人間による視覚障害者支援と同様の有益な情報を視覚障害者に提供できるシステム構築を実現した．

9.5　実用化までの課題

本研究で述べたスマート白杖の実用化までの課題を示す．

まず，超音波センサの取得距離の安定性が求められる．超音波センサのジャンプ処理で述べた通り，超音波センサを同時に複数個使用するため周期パルス信号を用いて同期を行ったが，意図しない取得距離を受信してしまう時があった．本研究では過去二つ分のデータと比較を行い，誤検知と判断した場合には一つ前のデータとの置き換えを行った．しかし，この方法では正確な取得距離も置換されてしまう可能性が考えられるため，超音波距離センサの取得距離の安定性の向上が必要である．

次に，左右側面方向の検知距離の使用方法である．左右側面方向の距離検出は視覚障害者の生活環境に大きく影響を受けるため，有益な情報であるかは各個人により異なる．そこで，本研究では検知範囲の追加（カスタマイズ）とし距離データの取得のみとしたが，具体的な情報提供を行えるよう，検知距離の判別方法の検討が必要である．

また，装置のサイズ，電池の持ちなどの検討も必要である．本研究では，視覚障害者の方が使用している白杖に外付けできることを目指した．試作品では白杖の重さ 140g に対し，装置の重さが 160g とし，この重さの中には電源として使用している角電池の 42g も含まれている．白杖への取り付け位置など，使用者が振りやすい重心に装着しているが，装置の軽量化は必須であると考える．また，電源として使用している角電池は 1 日の稼働で消耗してしまった．電池の持ちについても検討が必要である．

視覚障害者への通知方法の検討も必要である．本研究ではデータ処理部にスマートフォンを使用することにより装置自体の小型化を図ったため，視覚障害者への通知にも追加の機器を設置することなく，スマートフォン内蔵の音声出力と振動子を使用した．しかし，より確実に使用者へ通知を行うためには，骨伝導スピーカーなどの使用を検討する必要がある．

ユーザーインターフェイスについてはより簡単な操作が求められる．また，システムに異常が生じた場合や使用者が誤操作した場合などに備え，フェールセーフの考慮も必要であると考えられる．

最後に，福祉機器や福祉器具の開発において，研究レベルのものは多く存在するが，製品として世の中に出ない，製品化しても普及するものが少ないという現状があった．要因としては，開発者側と当事者の方との間でうまくコミュニケーションが取られずに使えないものや使いにくいものが製作されてしまうということがあった．そのため，視覚障害への理解を深め，また，ご意見を頂くために聞き取り調査や展示会などに出品し，様々なアドバイスを頂いた．しかし，視覚障害者の方が使いやすい製品にするためには，生活環境の異なる視覚障害者の方々に使用して頂き，意見を頂くことが必要であると感じた．

以上が，今後の課題である．

謝辞

本研究を進めるに当たり，終始懇切なるご指導を賜りました金丸隆志准教授に深く感謝の意を表します．また，本論文の執筆に当たり，丁寧かつ熱心なご指導を賜りました工学部機械システム工学科濱根洋人准教授，グローバルエンジニアリング学部機械創造工学科堀内邦雄准教授，産学連携 ECP センター花野井利之様に心より御礼申し上げます．

視覚障害者の現状調査にあたり快くご協力くださりました，茨城県立視覚障害者福祉センターの皆様に感謝の意を表します．また，いつも私を支えてくれた電子素子物理工学研究室の同輩，後輩の皆様に感謝致します．中でも，机を並べて共に 2 年間学び，研究だけではなく私生活においても良き友人である中込恭平氏，野崎祐基氏，山口和也氏にこの場で感謝の意を表します．

最後に，遠方からいつも私を支え，応援して下さった両親に感謝致します．

参考文献

[1] 厚生労働省, “障害者の方への施策,” 厚生労働省,
Available:http://www.mhlw.go.jp/seisakunitsuite/bunya/koyou_roudou/koyou/shougaishakoyou/shisaku/shougaisha/index.html.

[2] 厚生労働省社会・援護局障害保健福祉部企画課, “平成 18 年身体障害児・者実態調査結果,Available: http://www.mhlw.go.jp/toukei/saikin/hw/shintai/06/dl/01.pdf.

[3] 千葉県視覚障害者福祉協会, “視覚障害者理解のために,”
Available: http://www1.odn.ne.jp/tisikyo/sikaku.html.

[4] 厚生労働省, “平成 19 年国民健康・栄養調査報告,”
Available: http://www.mhlw.go.jp/bunya/kenkou/eiyou09/01.html.

[5] 日本眼科学会, “緑内障,”
Available: http://www.nichigan.or.jp/public/disease/ryokunai_ryokunai.jsp.

[6] 内閣府, “身体障害者福祉法,” 総務省法令データ提供システム,
Available: http://law.e-gov.go.jp/htmldata/S25/S25F03601000015.html.

[7] 国立障害者リハビリテーションセンター, “白杖について,” 国立障害者リハビリテーションセンター学院視覚障害学科,
Available: http://www.rehab.go.jp/College/japanese/yousei/rb/lnk_cane.html.

[8] 総務省, “総務省法令データ提供システム,”
Available: http://law.e-gov.go.jp/htmldata/S35/S35HO105.html.

[9] 秋田精工株式会社, “スマート電子白杖,” 秋田精工株式会社,
Available: http://www.akitaseiko.jp/.

[10] 秋田県, “視覚障害者用電子白杖購入助成事業,”
Available: http://www.pref.akita.lg.jp/www/contents/1306729826824/index.html.

[11] University of Konstanz, “Human-Computer Interaction,” University of Konstanz,
Available: http://hci.uni-konstanz.de/blog/2011/03/15/navi/?lang=en.

[12] L. Ciaffoni, “Ariadne GPS,” Ariadne GPS,
Available: http://www.ariadnegps.eu/tickets/.

[13] 国立障害者リハビリ研究所, “福祉工学カフェ　第二回視覚障害者支援関連,”
著: *福祉工学カフェ*, 新宿区, 2010.

[14] hrdakinori, “BT_DROID,”
Available: https://github.com/hrdakinori/BT_DROID#readme.

[15] BUFFALO, “Bluetooth 機器,” BUFFALO,
Available:http://buffalo.jp/products/catalog/supply/bluetooth/bluetooth/adapter/bshsbd03/.

[16] オーミック電子株式会社, “超音波センサの基本,”
Available: http://www.ohmic.jp/basics8.pdf.

[17] 竹上健, “視覚障害者のための超音波距離センサーによる周辺状況認知の基礎研究,”
埼玉学園大学紀要　第 10 号, 2010.

[18] MaxBotix Inc., “LV-MaxSonar-EZ1 Data Sheet,” MaxBotix Inc., USA, 2010.

[19] 株式会社秋月電子通商, “KXM52－1050 モジュール,” 株式会社秋月電子通商.

[20] NTT docomo, “らくらくスマートフォン,”
Available: http://www.nttdocomo.co.jp/product/easy_phone/f12d/index.html.

[21] 後閑哲也, 作って遊べるロボット工作, 技術評論社, 2003.

[22] 後閑哲也, C 言語ではじめる PIC24F 活用ガイドブック, 技術評論社, 2007.

[23] MICROCHIP, “PIC24FJ64GB004 Family Data Sheet,”
Available:http://ww1.microchip.com/downloads/en/devicedoc/39940c.pdf#search='PIC24Fj64gb004+datasheet'.

[24] 高木佑介, 金丸隆志, “Android OS とカメラによる対象物追跡における処理の高速化,”
工学院大学大学院　修士論文, 新宿区, 2012.

[25] 金丸隆志, フーリエ変換入門, ソフトバンククリエイティブ, 2011.

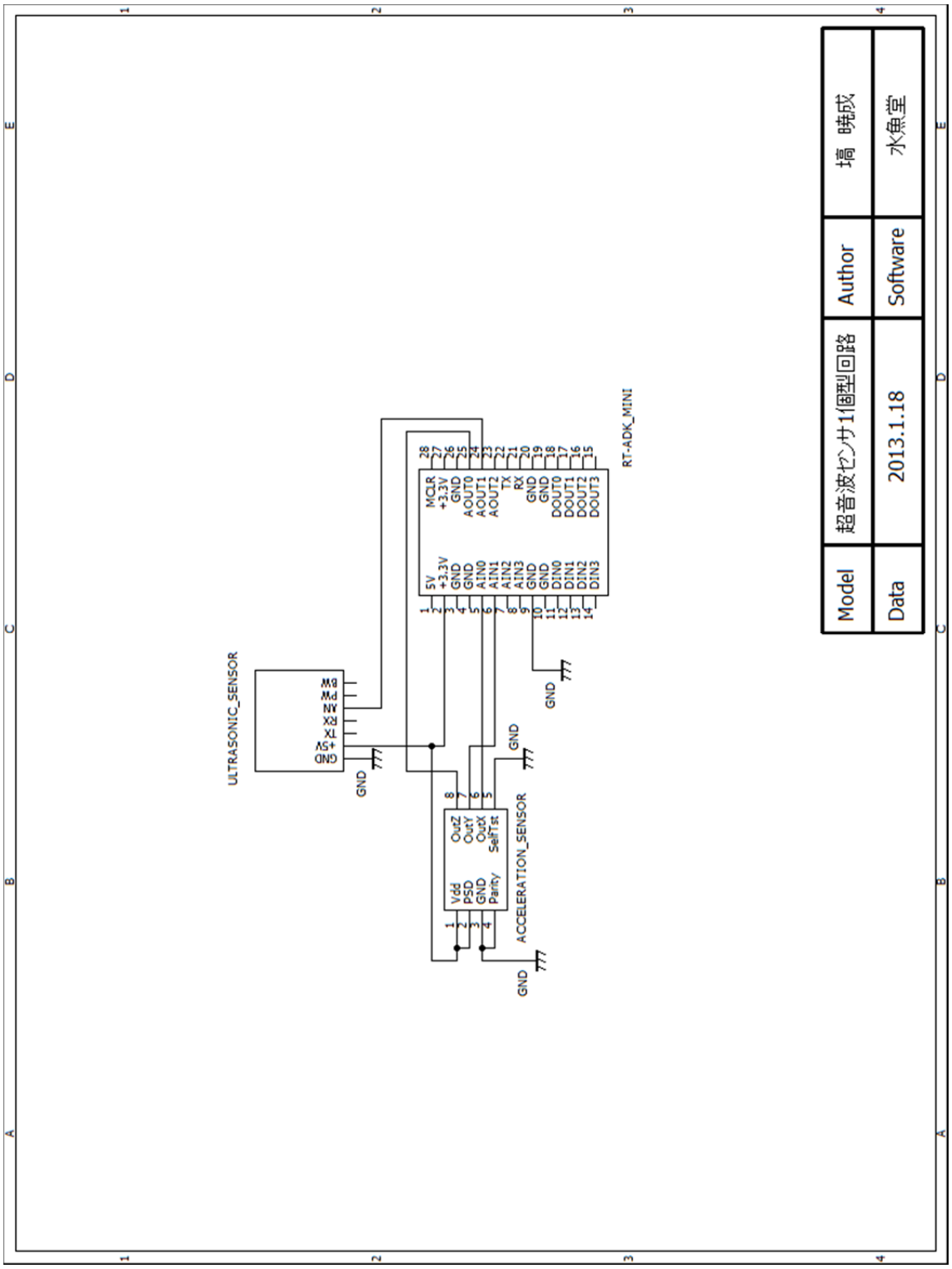
[26] B. W.H.Press, NUMERICAL RECIPES in C, 技術評論社, 1993.

[27] MaxBotix Inc., “Using Multiple MaxSonar Sensors,”
Available: http://www.maxbotix.com/articles/031.htm.

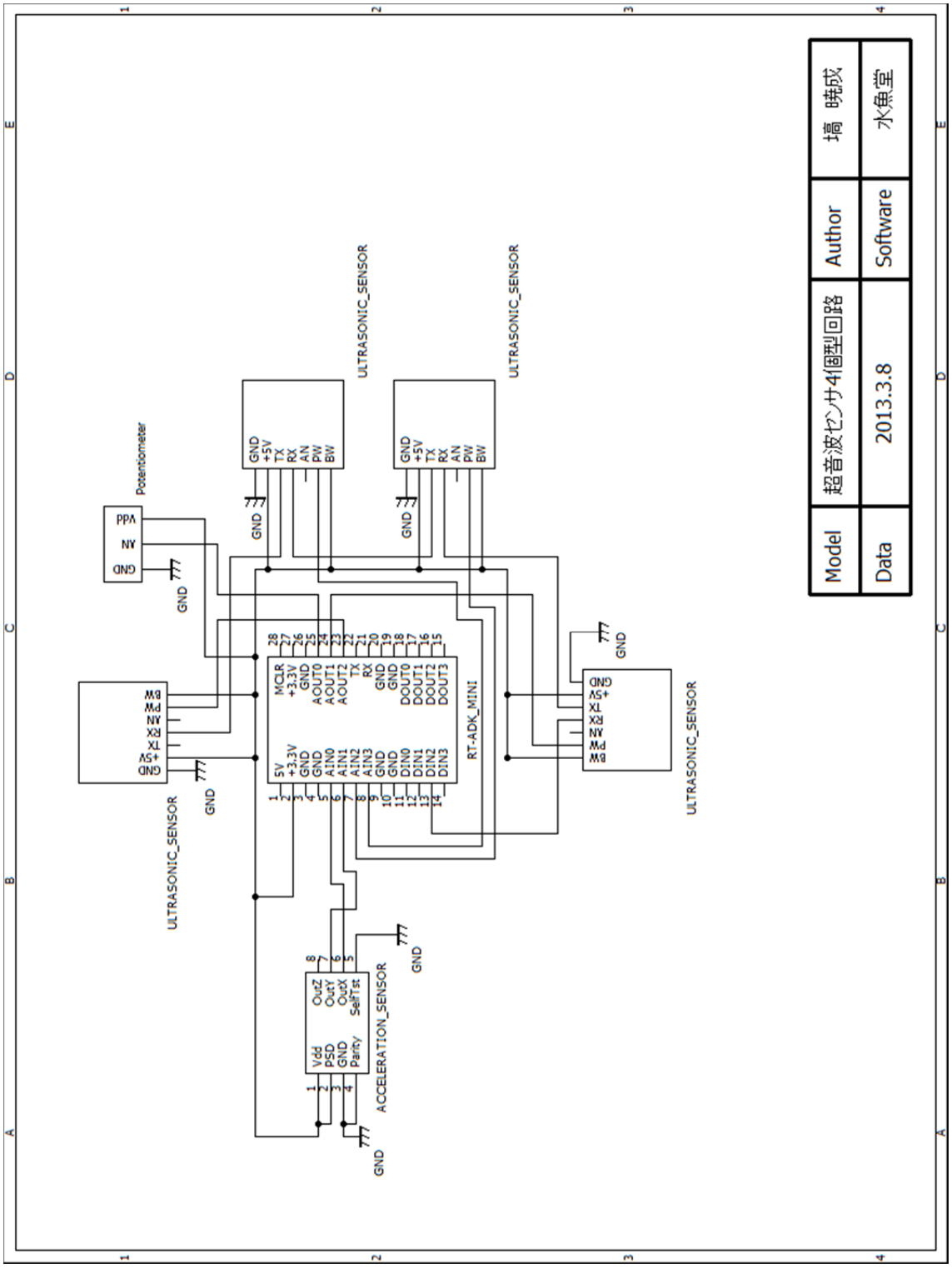
[28] TechBooster, “MediaPlayer で音楽を再生する,”
Available: http://techbooster.jpn.org/andriod/application/267/.

図表目次

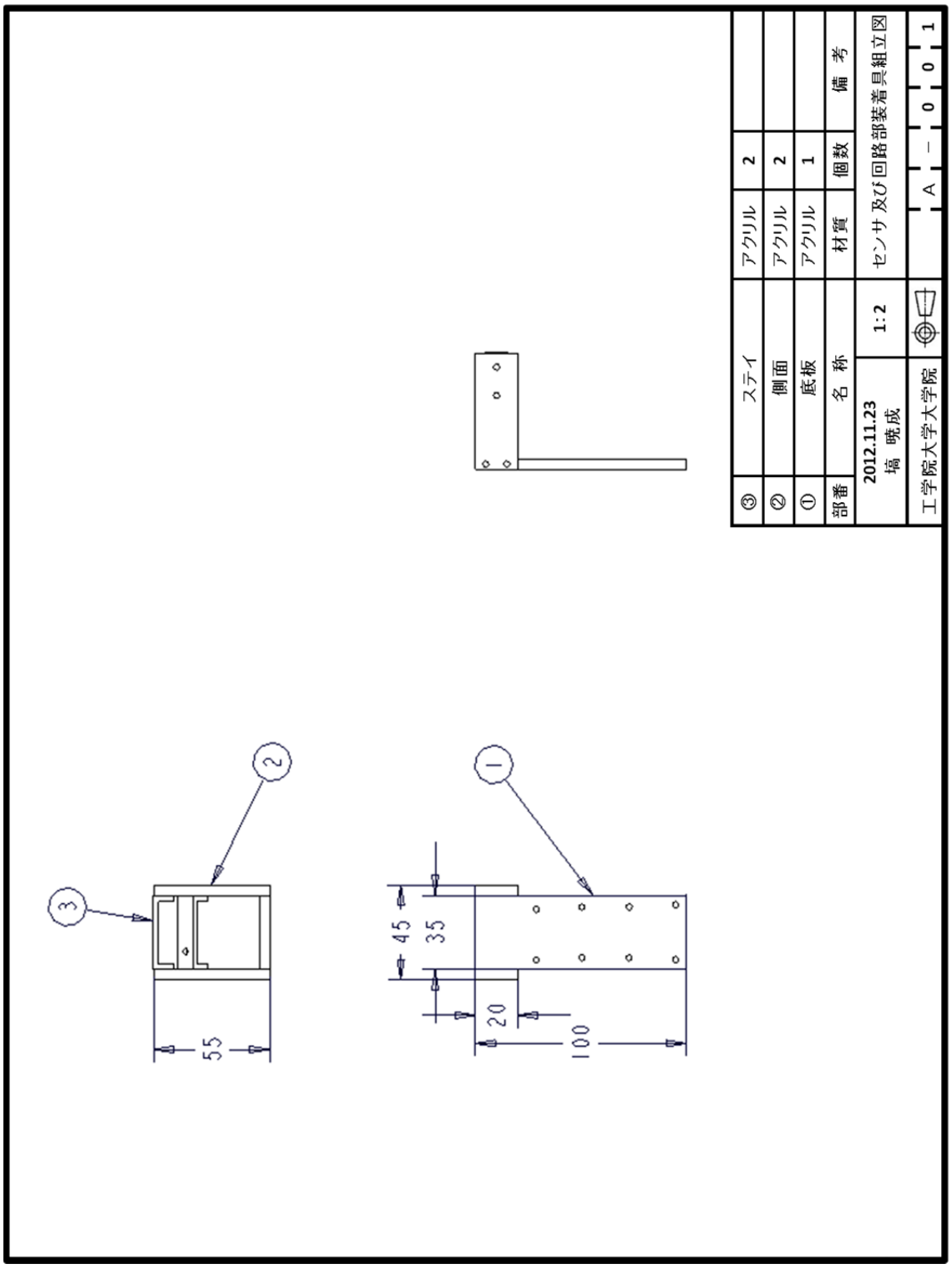
付録１　超音波センサ１個型回路図



付録２　超音波センサ４個型回路図



付録３　センサ及び回路部装着具組立図



付録４　JNI ソースコード

```
#include <jni.h>
#include <android/log.h>

#include "fft.h"
#include "mnewt.h"

#define   LOG_TAG      "DATA"
#define                                                                    LOGI(...)
__android_log_print(ANDROID_LOG_INFO,LOG_TAG,__VA_ARGS__)
#define                                                                    LOGE(...)
__android_log_print(ANDROID_LOG_ERROR,LOG_TAG,__VA_ARGS__)

void f_dfdx(float x[],int n,float fvec[],float fjac[MNEWT_N][MNEWT_N]);

float ssin, scos, sin2, cos2, sinxcos;
float param[MNEWT_N];

int stored_num = 0;
float  local_acc[NDATA];   // Java から貰った、大きなギャップを消した加速度データ
NDATA 個（32 個の予定）
float local_time[NDATA];    // データの時刻を NDATA 個（32 個の予定）
float local_data[NDATA];    // local_acc から平均値を引いたもの（こちらを主に用いる）
float local_sonic[NDATA];    // 超音波データ

// FFT 計算結果格納用
float fft_real[NDATA];
float fft_img[NDATA];
float fft_norm2[NDATA];

jfloat
Java_com_hanawa_android_DroidControl_DroidControlActivity_getEvaluationValue( J
NIEnv* env,jobject thiz,
                        jfloat jtime, jfloat jacc, jfloat jsonic);
```

```
jfloat
Java_com_hanawa_android_DroidControl_DroidControlActivity_getsdata(      JNIEnv*
env,jobject thiz);

void f_dfdx(float x[],int n,float fvec[],float fjac[MNEWT_N][MNEWT_N]);

jfloat
Java_com_hanawa_android_DroidControl_DroidControlActivity_getEvaluationValue( J
NIEnv* env,jobject thiz,
                        jfloat jtime, jfloat jacc, jfloat jsonic){

  // NDATA 個（32 個の予定）のデータを準備
  int i, j, k;
  int N = NDATA;

  for(j=0 ; j<N-1 ; j++){
    local_time[j] = local_time[j+1];
    local_acc[j] = local_acc[j+1];
    local_sonic[j] = local_sonic[j+1];

  }
  local_time[N-1] = jtime;
  local_acc[N-1] = jacc;
  local_sonic[N-1] = jsonic;

  if(stored_num<N){
    stored_num++;

    if(stored_num<N){
      return 0;
    }
  }

  //char buf[100];
  //LOGE("we got 32 data");
```

```
///// local_sdata の平均 s と標準偏差 s2 を計算
float s = 0;
float s2 = 0;

for(j=0 ; j<N ; j++){
   s += local_acc[j];
   s2 += local_acc[j]*local_acc[j];
}
s /= N;
s2 /= N;

s2 = s2 - s*s;
if(s2 > 0){
   s2 = sqrt(s2);
}else{
   s2 = 0;
}

//sprintf(buf,"s=%g, s2=%g¥n",s,s2);
//LOGE(buf);

// local_data は local_acc から平均値を引いたもの
for(j=0 ; j<N ; j++){
      local_data[j] = local_acc[j] - s;
}

// FFT 用の変数を初期化
for(j=0 ; j<N ; j++){
   fft_real[j] = local_data[j];
   fft_img[j] = 0;
}

// FFT 計算
fft(N, fft_real, fft_img);

//LOGE("FFT done");
```

```
// FFT のノルムの二乗を計算
for(j=0 ; j<N ; j++){
   fft_norm2[j] = fft_real[j]*fft_real[j] + fft_img[j]*fft_img[j];
}

//FFT スペクトルの最大値を決定
float maxval = -1;
int maxpos = 1;
for(j=1 ; j<N/2 ; j++){
   if(fft_norm2[j]> maxval){
      maxval = fft_norm2[j];
      maxpos = j;
   }
}

// 加速度の周波数を決定
float freq = 0;
int val = 0;
for(j=1 ; j<N/2 ; j++){
   if(fft_norm2[j]> maxval/2){
      freq += j/(local_time[N-1] - local_time[0]);
      val += 1;
   }
}
if(val==0){
   freq = 1;
}else{
   freq /= val;
}

//sprintf(buf,"freq=%g¥n",freq);
//LOGE(buf);

// 振幅 A (param[0]) と位相 θ (param[1])の初期値を決定
param[0] = s2; // 振幅 A の初期値は標準偏差にする
```

```
  float minval = 0;
  int minpos = 0;
  for(k=0 ; k<N ; k++){// ずらしながら、ベストな位相を探す
    float phase = k*2*M_PI/N;

    float sum =0;
    for(j=0 ; j<N ; j++){
      sum += fabs( local_data[j]- param[0]*sin(2*M_PI*freq*local_time[j] +
phase ) );
    }

    if(k==0){
      minval =sum;
    }else{
      if(sum < minval){
        minval = sum;
        minpos = k;
      }
    }
  }
  param[1] = minpos*2*M_PI/N;

  //sprintf(buf,"A=%g, theta=%g¥n",param[0], param[1]);
  //LOGE(buf);

  ///// ニュートン法でベストな A と θ を決める
  ssin = scos = sin2 = cos2 = sinxcos = 0;
  for(j=0 ; j<N ; j++){
    float sinval = sin(2*M_PI*freq*local_time[j]);
    float cosval = cos(2*M_PI*freq*local_time[j]);
    ssin += local_data[j]*sinval;
    scos += local_data[j]*cosval;
    sin2 += sinval*sinval;
    cos2 += cosval*cosval;
    sinxcos += sinval*cosval;
  }
```

```
  ssin /= N;
  scos /= N;
  sin2 /= N;
  cos2 /= N;
  sinxcos /= N;

  int ntrial = 100;

  float tolx = 0.0001;
  float tolf = 0.0001;

  // 初期値をバックアップ
  float bak0   = param[0];
  float bak1   = param[1];

  // ニュートン法
  int ret = mnewt(ntrial, param, 2, tolx, tolf);

  // エラーが出たら、初期値で妥協する
  if(ret<0){
    param[0] = bak0;
    param[1] = bak1;
  }

  float summed_val = 0;

  float period = 1/freq;
  for(j=N-1 ; j>=1 ; j--){

    if(local_time[N-1]-local_time[j] > period){
        break;
    }

    summed_val += local_sonic[j]*param[0]*sin(2*M_PI*freq*local_time[j] +param[1])
* (local_time[j]-local_time[j-1]);
  }
```

```
    summed_val /= period;
    return summed_val;

}

jfloat
Java_com_hanawa_android_DroidControl_DroidControlActivity_getsdata(      JNIEnv*
env,jobject thiz){
      return    local_data[NDATA-1];
}

void f_dfdx(float x[],int n,float fvec[],float fjac[MNEWT_N][MNEWT_N]){

    float sintheta = sin(x[1]);
    float costheta = cos(x[1]);
    float sintheta2 = sintheta*sintheta;
    float costheta2 = costheta*costheta;
    float sincostheta = sintheta*costheta;

    float val1 = costheta*ssin + sintheta*scos;
    float val2 = costheta2*sin2 + sintheta2*cos2 + 2*sincostheta*sinxcos;
    float val3 = costheta*scos - sintheta*ssin;
    float val4 = sincostheta*(-sin2+cos2) + (costheta2-sintheta2)*sinxcos;

    fvec[0] = -val1 +x[0]*val2;

    fvec[1] = -x[0]*val3 + x[0]*x[0]*val4;

    fjac[0][0] = val2;
    fjac[0][1] = -val3+2*x[0]*val4;
    fjac[1][0] = -val3+2*x[0]*val4;
    fjac[1][1] = x[0]*val1
                    +x[0]*x[0]*((costheta2-sintheta2)*(cos2-sin2)
                                    -4*sincostheta*sinxcos );

}
```