

卒業論文 2010 年度 (平成 22 年度)

ZigBee ノードの自動較正測位手法

慶應義塾大学
環境情報学部

学籍番号: 70746629

氏名: 広石 達也

2011 年 1 月 19 日

ZigBee ノードの自動較正測位手法

センサを使って収集した実空間情報を情報空間で共用し，利活用する考え方やそれに基づく情報システムは，ユビキタスコンピューティングや Internet of Things と呼ばれている．こうした情報システムではアプリケーションに応じて様々なセンサ等が使われるが，どのようなセンサデバイスであっても，測定する対象物や対象者の位置情報の取得が必要不可欠である．測位方法として広く使われている GNSS(Global Navigation Satellite System) は，衛星が見通し外になってしまう屋内での使用が原理的に難しい．このため，GNSS を補完する様々な屋内位置情報システムが提案されている．

本論文では，その中でも低消費電力，マルチホップ，アプリケーション層まで標準化されていて，センサネットワークとしても機能する ZigBee を用いた測位システムを提案し，その性能を評価した結果を報告する．ZigBee を用いた屋内測位システムは，受信ノードが受信した信号の電波強度 (RSSI) を測定し，その値から送信ノードとの距離を求め，三角測量によって位置を推定する．このため RSSI と距離の関係を正確に導くことが求められる．一般的に，この 2 つは対数関係にあると考えられているため，RSSI から距離を求める際には対数計算を行う必要がある．これが ZigBee チップには厳しい要件となっていた．本論文では，対数計算を線形近似したとしても，測位を実施する際の参照点の数を 6 箇所程度にすることで，測定誤差を損なうことなく対数計算を一次近似できることを明らかにする．これにより，RSSI から距離を求める計算が簡単になり，250kB のスタックメモリがあれば，3 次元非線形の測位計算をソフトウェアのみで実現できることを示す．

さらに，参照点周辺であらかじめ測定しておいた RSSI と距離の関係を，測位の際に参照点から測定対象の ZigBee ノードに通知するプロトコルである ACPS(Automatic Calibration Positioning System) を提案する．ACPS による効果は実験によって確認し，3 次元測定において 13.4% の測定精度向上が実現できることを明らかにする．

キーワード

1: 測位, 2: ZigBee, 3: ソフトウェア, 4: 自動較正, 5: RSSI

慶應義塾大学 環境情報学部
広石 達也

Automatic Calibration Positioning System by ZigBee Nodes

Recently, real space information is collected via sensor network and shared in virtual space, such as environment called Ubiquitous Computing or Internet of Things. Location information is indispensable when using sensor data. GNSS(Global Navigation Satellite System), which is commonly used for positioning in outdoor environments, cannot be used in indoor environments because of unreachable radio waves. There are various positioning systems for indoor localization accordingly.

In this thesis, I propose an indoor positioning method using ZigBee, and report the result of performance evaluation. ZigBee is a low-cost, low-power, wireless mesh networking standard, and also has a function of sensor network. Indoor positioning system by ZigBee calculates distance between sending and receiving nodes using RSSI value of message packets, and estimates its position by a triangulation. It is important to find accurate logarithmic relationship between Received Signal Strength Indicator(RSSI) and the distance. Using sufficient reference points, about 6 nodes, positioning accuracy can be maintained even if using linear approximation instead of nonlinear formula. Linear approximation makes it easier to implement software-based 3D positioning system in a low-power ZigBee chip.

In addition, I propose ACPS(Automatic Calibration Positioning System) protocol, where each reference point calculates its own parameters which show appropriate relationship between RSSI and distance used in positioning, and sends to target node. I show that ACPS improves 13.4% of 3D positioning accuracy by experiments.

Keywords

1: Positioning System, 2: ZigBee, 3: Software, 4: Automatic Calibration, 5: RSSI

Keio University, Faculty of Environment and Information Studies
Tatsuya HIROISHI

目次

第 1 章	序論	1
1.1	背景	1
1.1.1	情報環境の広がり	1
1.1.2	位置情報の意義	1
1.1.3	測位技術の使い分け	2
1.2	研究の目的	3
1.3	本論文の構成	4
第 2 章	測位システム	5
2.1	測位技術	5
2.1.1	GNSS	6
2.1.2	IMES	6
2.1.3	無線 LAN 測位	7
2.1.4	Bluetooth	7
2.1.5	UWB	7
2.1.6	RFID	8
2.1.7	超音波測位	8
2.1.8	光を利用した測位	8
2.2	ZigBee を用いた測位	8
2.2.1	ZigBee	9
2.2.2	Location Engine	10
2.3	まとめ	14

第 3 章	Location Engine の課題	16
3.1	既存研究	16
3.1.1	Location Engine を用いたヒトの移動検出に関する研究	16
3.1.2	移動する ZigBee センサーノードの位置推定アルゴリズム	17
3.1.3	ZigBee Location Profile における移動センサノードの省電力化プロトコル	17
3.2	ZigBee ノードの通信距離と RSSI の関係	17
3.2.1	目的	17
3.2.2	測定概要	18
3.2.3	結果	18
3.3	Reference Node の個数と測位精度	19
3.3.1	目的	20
3.3.2	実験概要	20
3.3.3	結果	20
3.4	まとめ	22
第 4 章	Location Engine 改善へのアプローチ	26
4.1	距離導出の計算	26
4.1.1	ソフトウェア実装	26
4.1.2	距離計算の線形近似	26
4.1.3	測位計算アルゴリズム	28
4.2	パラメータの自動校正	29
4.2.1	Reference Node 毎の最適パラメータ	30
4.2.2	システム構成	30
4.3	まとめ	32
第 5 章	改良版測位システムの評価	33
5.1	測位システムのソフトウェア実装	33
5.2	自動校正測位のシミュレーション	34
5.2.1	ACPS の実装	34
5.2.2	測位シミュレーション実験	36

5.2.3	測位誤差の比較	36
5.3	まとめ	39
第 6 章	結論と今後の展望	40
6.1	結論	40
6.2	今後の展望	41
6.2.1	移動時の測位	41
6.2.2	RSS 方式以外の測位	41
付録 A	LDL^T 分解と前進・後退代入法	42
A.1	LDL^T 分解	42
A.2	前進・後退代入法	43
付録 B	測位システムソースコード	45
謝辞		48
参考文献		49

図目次

1.1	Internet of Things のイメージ	2
2.1	ToA/RSS 方式の測位	6
2.2	TDoA 方式の測位	6
2.3	AoA 方式の測位	6
2.4	ZigBee の構成	9
2.5	ZigBee の使用例	10
2.6	Location Engine が構成するネットワーク	11
2.7	ZigBee BB	12
2.8	ZigBee EB	12
2.9	Location Engine メッセージシーケンス	13
2.10	Location Engine デモの様子	14
3.1	RSSI 測定実験	18
3.2	RSSI 測定実験の様子	18
3.3	RSSI と距離の関係	19
3.4	Reference Node の配置図	21
3.5	実験室内における各ノードの配置図	21
3.6	測位実験の様子	22
3.7	Reference Node の個数毎の測位誤差分布図	23
3.8	Reference Node の個数毎の測位誤差を示す箱ひげ図	23
3.9	Reference Node の個数毎の測位誤差ヒストグラム (1)	24
3.10	Reference Node の個数毎の測位誤差ヒストグラム (2)	25

4.1	距離と RSSI の関係と近似式	27
4.2	従来版 Location Engine の XY-RSSI Response パケット	31
4.3	改良版測位システムの XY-RSSI Response パケット	31
4.4	改良版測位システムのメッセージシーケンス	31
5.1	Reference Node の配置図 (再掲)	37
5.2	実験室内における各ノードの配置図 (再掲)	37
5.3	2 次元測位の誤差	38
5.4	3 次元測位の誤差	38

表目次

3.1	Reference Node 毎の最適パラメータ	19
4.1	線形近似と非線形近似の測位誤差 (m)	28
5.1	開発環境	33
5.2	Blind Node ファイルサイズの比較	34
5.3	測位システムの特徴	35
5.4	測位誤差の比較 (m)	39

第 1 章

序論

本章では，本論文の背景，目的について述べる．また，本論文の構成を示す．

1.1 背景

本節では，本論文の背景について述べる．

1.1.1 情報環境の広がり

環境に埋め込んだセンサを使って収集した実空間情報をネットワーク上で共用し，利活用するという考え方やそれに基づく情報システムが広がっている．このような情報環境を，ユビキタスコンピューティングや Internet of Things と呼ぶ．ユビキタスコンピューティングは，いつでも，どこでも，だれでも利用できるコンピューティング環境のことで，コンピュータの存在を意識させずに人を支援する技術である．Internet of Things は，日本語では「モノのインターネット」と訳され，センサや RFID によってモノ同士がつながる世界である (図 1.1)．近年，センサネットワークを含む情報環境を前提とし，人を支援するようなシステムが注目されている．

1.1.2 位置情報の意義

コンピュータやネットワークが様々な所に存在する環境においてサービスを提供するためには，対象者や対象物の位置情報の取得が必要不可欠である．



図 1.1 Internet of Things のイメージ

現在，位置情報を用いた様々なサービスが存在する．自動車の位置情報を用いたカーナビゲーションシステムでは，目的地までの経路を表示して案内する．また，携帯電話の位置情報によって，紛失時にどこで落としたのかを調べることができる．防犯ブザーの位置情報を利用した，小学生の通学見守りサービスも存在する．さらに，位置情報を利用した陣取り合戦のようなゲームも実現されている．

これらのサービスは，位置情報の取得なしでは実現できない．これからの社会を支える上で，いつでもどこでもだれでも位置情報サービスを利用できる環境が重要である．

1.1.3 測位技術の使い分け

位置情報の取得には，様々な測位技術が利用されている．

屋外における測位は GNSS(Global Navigation Satellite System) を用いるのが一般的である．しかし，屋内における測位では，衛星からの死角となるため，GNSS を利用することができない．そのため，他の技術を利用して GNSS を補完することが必要となる．情報環境の広がりによって無線 LAN のアクセスポイントの利用可能領域が広がっているため，屋内では無線 LAN を用いた測位が一般的に用いられている．また，UWB を用いれば，高精度の測位が実現できるが，デバイスの同期が必要となる．ZigBee はアプリケーション以外の部分が標準化されているため，開発が容易である．ZigBee の通信速度は低速であるが，マルチホップネットワークのため拡張性が高い．

これらの測位技術は，使用環境によって使い分けられる場合が多い．

1.2 研究の目的

本論文では、ZigBee を用いた測位に着目した。ZigBee の特徴として、低消費電力であること、センサネットワークとの親和性が高いこと、標準化が進んでいること、安価であることなどが挙げられる。このため、ZigBee を用いた測位は、配置コストを低く抑えたい場合に利用されると考えられる。

ZigBee を用いた測位システムとして、Texas Instruments が開発した Location Engine がある。Location Engine は、ZigBee チップが発する電波の強度 (RSSI: Received Signal Strength Indicator) を測定し、その減衰量から送信側と受信側の距離を求め、三角測量によって位置を推定する。よって、RSSI と距離の関係を正しく求めることが重要である。一般的に、RSSI と距離は対数関係にあると考えられているため、距離の導出には対数計算を用いる必要がある。Location Engine では距離算出の機構を、実装対象である CC2431 の内部に保持している。よって Location Engine は CC2431 のレジスタ実装に依存しているため、CC2431 以外では使用できない。また、Location Engine は 2 次元測位で実装されているため、高さを考慮した 3 次元測位では利用できない。そして、利用環境によって電波の伝搬が異なるため、RSSI と距離の関係を表すパラメータを手動で設定しなければならない。

本論文では、Location Engine をもとに、ソフトウェアベースの ZigBee 測位システムを提案する。距離算出における対数計算を線形近似しても、十分な数の参照点が存在すれば、測位精度を損なうことなく位置を求めることができることを示す。それによって、RSSI から距離を求める計算の過程が簡単になり、ソフトウェアベースで測位システムが実装可能になるため、CC2431 以外でも ZigBee の測位システムの利用できるようになる。また、3 次元の測位に対応することで、高さを考慮した測位が可能になる。さらに、参照点となる複数のノード同士が通信を行い、周辺の環境における RSSI と距離の関係を示すパラメータを算出し、測位対象の ZigBee ノードに通知するプロトコルである、ACPS(Automatic Calibration Positioning System) を提案する。

1.3 本論文の構成

本論文は 6 章から構成される。第 2 章では、既存の測位技術についてまとめ、ZigBee を用いた測位について述べる。第 3 章では、ZigBee の測位ソリューションである Location Engine の既存研究と、その特徴と課題について述べる。第 4 章では、Location Engine の改善点について述べ、改善案のアプローチを示す。第 5 章では、Location Engine を改良して実装した ACPS の評価を行う。第 6 章では、本論文の結論を述べ、今後の展望を示す。

第2章

測位システム

測位技術には，GPS，無線 LAN，UWB など，様々なシステムが存在する．これらの技術は，使用する環境に応じて使い分けられている．本章では，既存の測位技術についてまとめ，特に ZigBee を用いた測位について述べる．

2.1 測位技術

本節では，既存の測位技術についてまとめる．測位手法は大きく分けて以下の4つが挙げられる．

ToA: Time of Arrival

信号の到達時間から送信機と受信機の距離を測定する手法 (図 2.1)

TDoA: Time Difference of Arrival

信号の到達時間の差から送信機と受信機の距離の比を測定する手法 (図 2.2)

RSS: Received Signal Strength

信号強度の減衰量から送信機と受信機の距離を測定する手法 (図 2.1)

AoA: Angle of Arrival

信号の到来角度から位置を測定する手法 (図 2.3)

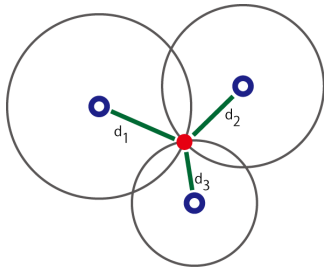


図 2.1 ToA/RSS 方式の測位

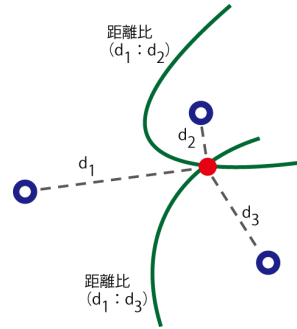


図 2.2 TDoA 方式の測位

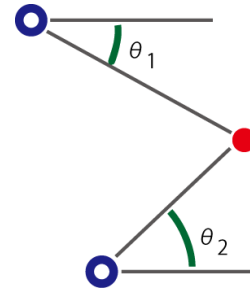


図 2.3 AoA 方式の測位

2.1.1 GNSS

GNSS(Global Navigation Satellite System)[1] とは、人工衛星を用いた測位システムで、人工衛星が発信する電波を利用して受信機の位置を測定する。よって GNSS は、人工衛星からの電波が受信できる場所なら地球上全体での利用が可能である。このシステムでは、ToA 方式の測位で緯度・経度・高度の位置情報を得る。また、TDoA 方式で位置情報を算出する研究も行われている [2]。測位誤差は、単独測位だと約 10m、DGPS や RTK を利用すると数 m となる。

代表的な GNSS として、米国が運用している GPS(Global Positioning System)、ロシアが運用している GLONASS(Global Navigation Satellite System)、EU が構築中の Galileo が挙げられる。日本でも、GPS の補正技術として準天頂衛星システム (QZSS: Quasi-Zenith Satellite System) がある。

屋外における測位では、GPS が最も普及している測位システムである。しかし、人工衛星からの電波が受信できない屋内では使用できない。

2.1.2 IMES

IMES(Indoor Messaging System)[3] とは、宇宙航空研究開発機構 (JAXA) が考案した屋内測位技術の一つである。IMES は、GPS の電波が届かない屋内に送信機を配置し、GPS や QZSS と互換性のある信号を用いる。GPS 受信機のファームウェアをアップデートすれば同一の受信機をそのまま利用することができることから、GPS と併用して屋内

外を通したシームレスな測位の実現が期待される。

2.1.3 無線 LAN 測位

無線 LAN を用いた測位システムでは、アクセスポイントから発信される電波を用いて無線 LAN 端末の位置を測定する。無線 LAN 測位は、既存の無線 LAN 端末をそのまま利用できる利点がある。測位の方法は、ToA/TDoA 方式、RSS 方式、観測地点における信号強度を登録したデータベースを参照する方法が挙げられる。Place Engine[4] では、RSS 方式で位置測定を行う。また、Locky.jp[5] では、アクセスポイントの ID と観測地点における RSS を登録したデータベースを用いて位置情報サービスを提供している。

2.1.4 Bluetooth

Bluetooth は、電子機器用の近距離無線通信規格の 1 つであり、主に PC の周辺機器や携帯電話などのモバイル端末に搭載されている。無線接続状態を意識せずに常時接続したままで、文字情報や音声情報といった低速度のデジタル情報を使用する状況に適している。測位においては、無線 LAN 測位と同様に RSS 方式の測位が考えられる [6]。

2.1.5 UWB

UWB(Ultra Wide Band) とは、近距離で高速通信が可能な超広帯域無線技術のことで、非常に広い周波数にわたって非常に小さい電力スペクトラム密度 (周波数あたりの電力) の電波を使用する。周波数帯に対する使用電力は極めて小さくノイズレベル以下であることから、既存の電波利用に悪影響を与えることなく広い周波数帯を利用できる。

UWB を用いた測位 [7][8][9] では、ToA/TDoA 方式や RSS 方式の他に AoA 方式で位置を特定する方法もある。UWB 測位では端末の同期が必要であるが、測位誤差は数 cm 程度と他の測位技術と比べると非常に小さく、消費電力も小さい。このことから、高精度測位技術として注目されている。

2.1.6 RFID

RFID(Radio Frequency Identification) とは、ID 情報を保持している RF タグと無線通信によって情報のやり取りをする技術である。電子マネーや身分証明証に用いられているような非接触 IC カードも広義の RFID に含まれる。RF タグには、リーダからの電波をエネルギー源として動作するパッシブタグ、電源を内蔵しているアクティブタグ、両者の特徴を組み合わせたセミパッシブタグの 3 種類がある。RFID は主に物流におけるサプライチェーン・マネジメント、図書館の物品管理、センサネットワークなどに用いられる。EPCglobal[10] では、RFID とネットワークを組み合わせることで包括的な物流コードの標準化を進めている。

測位においては、リーダ機能を持つ端末が RF タグを認識することで近接情報が得られる。複数の RF タグを用いて建造物への出入りを判別することや、GPS や無線 LAN 測位と組み合わせて用いられる場合もある [7]。

2.1.7 超音波測位

超音波を用いた測位では、音波の到来時間を用いて測位を行う。超音波測位では多数の超音波センサが必要になるが、測位誤差は数 cm 程度である。超音波測位システムの代表である Active Bat[11] では、移動端末から発信される超音波を天井に配置された複数の超音波レシーバで受信し、ToA/TDoA 方式で測位する。

2.1.8 光を利用した測位

光を用いた測位では、LED のような可視光や赤外線、レーザを用いる方法がある。発光パターンをカメラで認識して位置を測定する手法や、レーザの反射光をスキャンすることで位置を測定する手法が挙げられる [12]。

2.2 ZigBee を用いた測位

本節では、ZigBee を用いた測位について述べる。

2.2.1 ZigBee

ZigBee とは、ZigBee Alliance によって仕様の策定が行われている無線通信規格である。ZigBee の構成の内、APS 層と NWK 層は ZigBee Alliance が標準化し、PHY 層と MAC 層は IEEE802.15.4 を使用している。よって、ZigBee の使用者は APL 層を作成するだけでよい (図 2.4)。

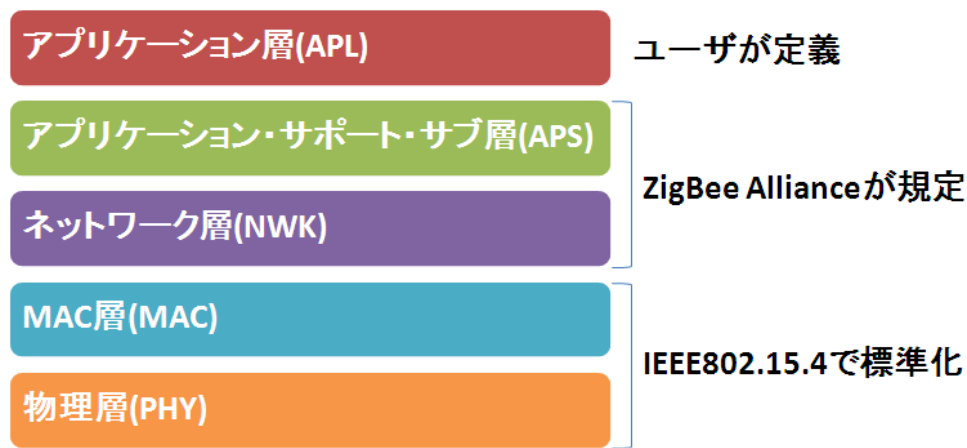


図 2.4 ZigBee の構成

データ転送速度は 20Kbps～250Kbps と低速で転送距離が短い、安価で消費電力が少ないといった特徴を持つ。このことから、家電や各種センサと組み合わせてホームオートメーションやセンサネットワークといったインフラへの応用が期待されている (図 2.5)。

ZigBee のネットワークは以下の 3 種類のノードによって構成される。

ZigBee Coordinator

ネットワーク内に 1 つ存在し、ネットワーク全体の制御を行う端末。

ZigBee Router

ネットワーク内でデータの中継機能を含む端末。

ZigBee End Device

ネットワーク内でデータの中継機能を持たない端末。

上記のノードがアドホックなマルチホップネットワークを構成する。

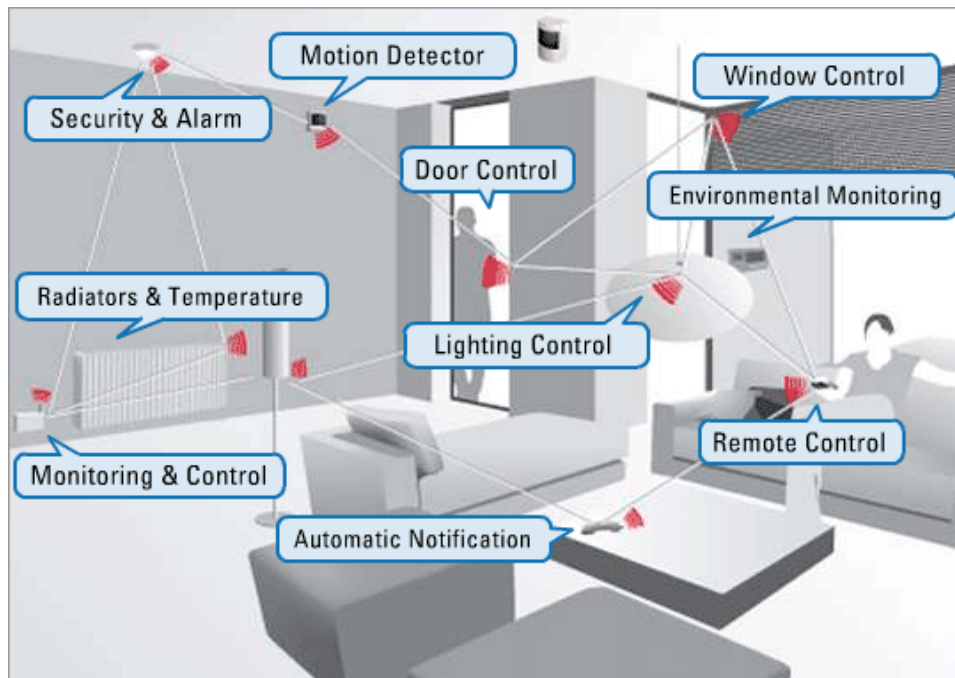


図 2.5 ZigBee の使用例

測位システムに ZigBee を使用する場合、各ノードが送信するパケットの RSSI を用いて測位を行う。ノードを配置すればマルチホップネットワークが構成されるため、屋内における測位や、展示会など手早く配置することが必要な状況で利用される。

2.2.2 Location Engine

ZigBee を用いた測位システムには、RSS 方式の測位 [13][14] と、RSSI 値をマッピングしてデータベースを参照する測位 [15] が存在する。Location Engine は Texas Instruments が開発した RSS 方式の ZigBee 測位システムである。Location Engine は、ZigBee 用 RFIC である CC2431 において実装される (図 2.6)。

ノード

Location Engine では、Reference Node、Blind Node、Location Dongle の 3 種類のノードを用いる。

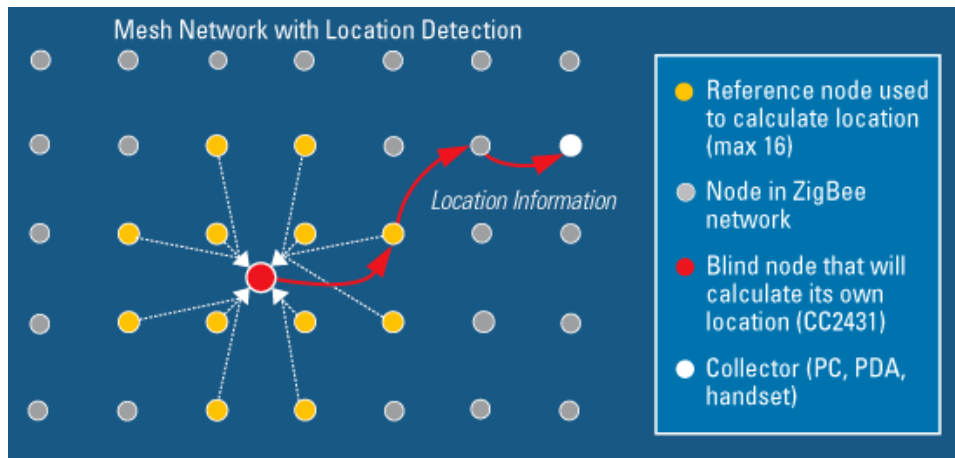


図 2.6 Location Engine が構成するネットワーク

Reference Node

ZigBee Router の機能を持ち，定位置に配置されたノード．自身の位置情報と Blind Node からの電波の強度を Blind Node に通知する．ZigBee BB(図 2.7) で動作する．

Blind Node

ZigBee End Device の機能を持ち，Reference Node との通信より，位置情報と電界強度情報を受信する．その情報から，ロケーションエンジンを通して自身の位置情報を算出し，Location Dongle に通知する．ZigBee BB で動作する．

Location Dongle

ZigBee Coordinator の機能を持ち，Reference Node, Blind Node のパラメータの更新や情報を管理する．ZigBee EB(図 2.8) で動作する．

メッセージシーケンス

Location Engine におけるノード同士のメッセージのやり取りは図 2.9 の通りである．各メッセージとノードの動作の詳細を以下に示す．

1. Location Dongle が Reference Node に対して XY 座標等の設定情報をパラメータとして付与して送信する (Reference Node Configuration).

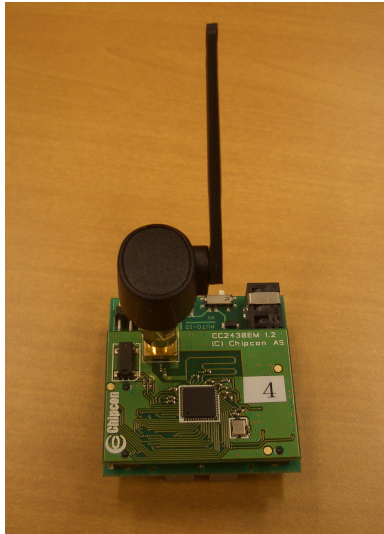


図 2.7 ZigBee BB

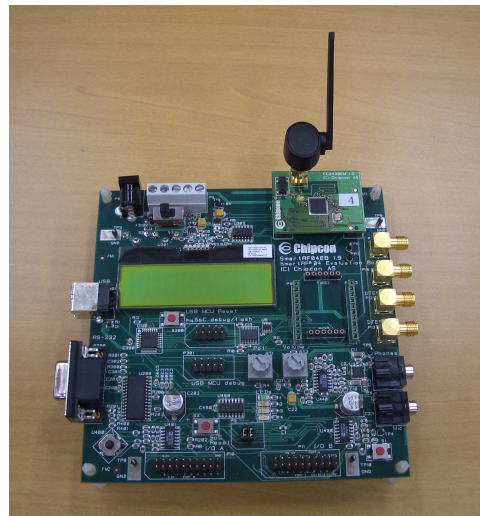


図 2.8 ZigBee EB

2. Location Dongle が Blind Node に対して Location Engine に必要な各種設定情報をパラメータとして付与して送信する (Blind Node Configuration).
3. Blind Node から各 Reference Node に向けてブロードキャストメッセージ (RSSI Blast) を送信する. Reference Node はメッセージの RSSI 値を算出して保持する.
4. Blind Node から各 Reference Node に向けてブロードキャストメッセージ (XY-RSSI Request) を送信する.
5. Reference Node が XY-RSSI Request のレスポンスとして, Blind Node に対し自身の XY 座標と RSSI Blast の RSSI 値をパラメータとして付与して送信する (XY-RSSI Response).
6. Blind Node が XY-RSSI Response をもとに自身の位置情報を算出する.
7. Blind Node 自身が算出した位置情報等をパラメータとして付与し, Location Dongle に向けて送信する.
8. 以降 3 から 7 を定期的に繰り返す.

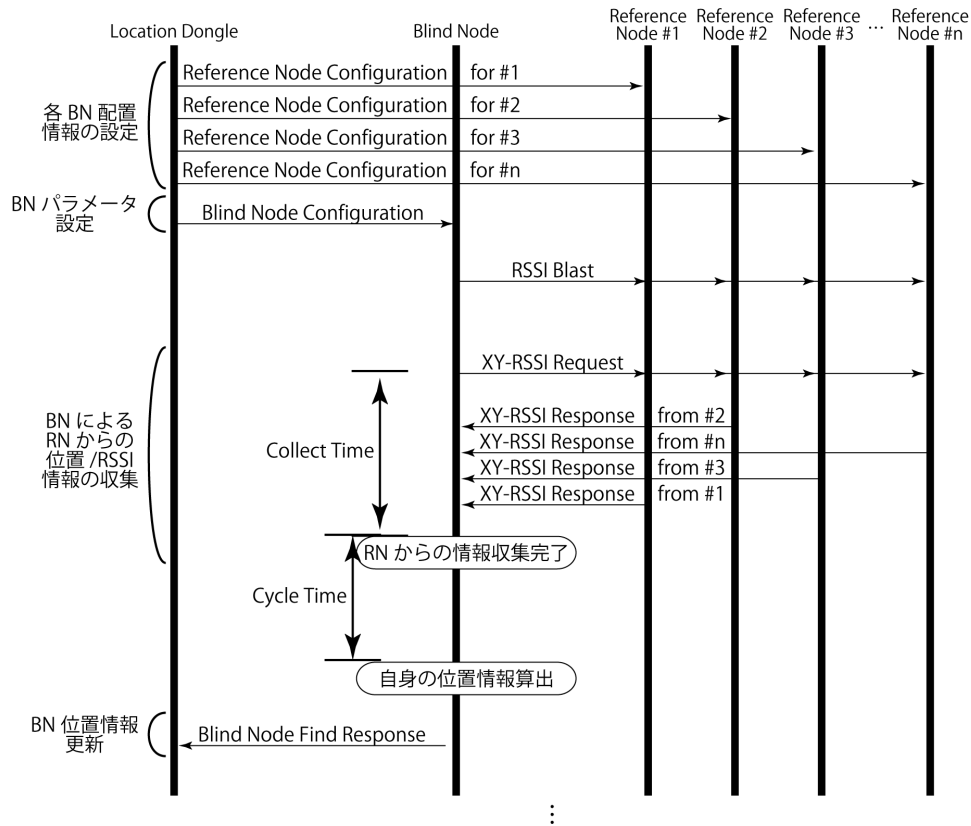


図 2.9 Location Engine メッセージシーケンス

位置算出

Location Engine は、電波の RSSI 値が伝搬距離に応じて減衰することを利用して距離計算を行っている。ZigBee の 2.4GHz 帯における RSSI 値と距離の関係は、式 (2.1) のように表される。

$$RSSI = -(10N \log_{10} d + A) \quad (2.1)$$

- A 1m 離れたところでの電界強度 (- dBm)
- N 経路品質係数
- d 通信距離 (m)

距離計算において、読み出した RSSI(-nn dBm) はそのまま式 (2.1) に適用される。パラメータ A と N は、Location Engine を使用する環境の中で実際に 1m, 3m, 5m と距離を変えて RSSI を測定し、Blind Node に設定する。

Location Engine は、電波の届く範囲に少なくとも 3 つの Reference Node があると Blind Node の XY 座標を算出することができる。Blind Node は各 Reference Node との距離を RSSI Blast で受信した RSSI と、あらかじめ設定してあるパラメータ A, N より計算し、自身の座標を算出する。Location Engine では、0.25m 単位で位置座標を求めることができる。Location Engine の測位アプリケーションを使用したデモの様子を図 2.10 に示す。

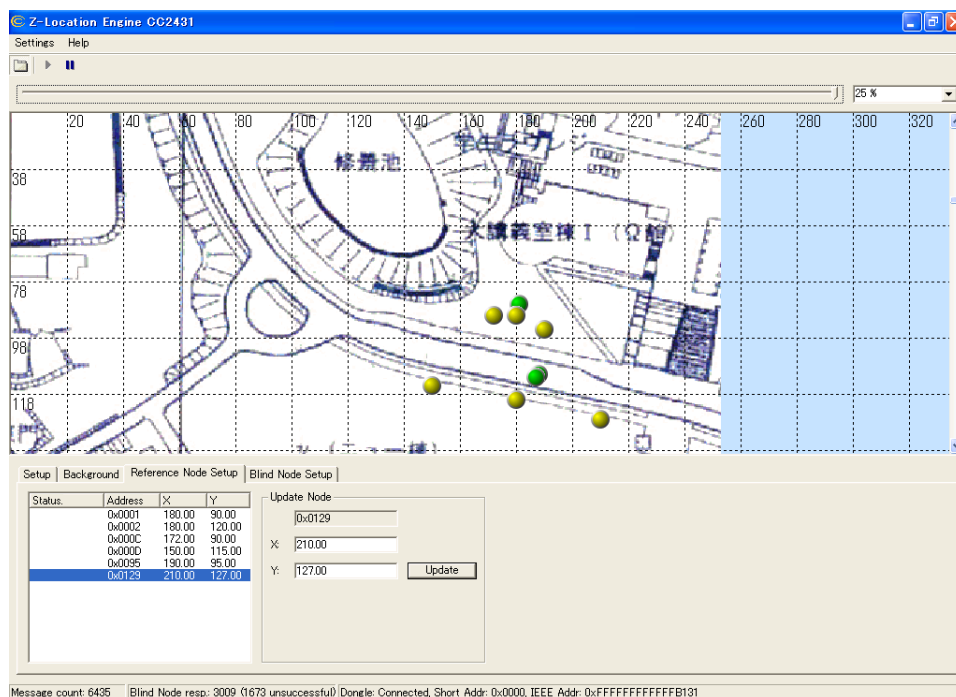


図 2.10 Location Engine デモの様子

2.3 まとめ

様々な測位技術が存在し、環境や条件によって使い分けられている。屋外における測位では GNSS を用いれば良いが、屋内における測位ではこれといって決め手となる測位シ

システムが選べないのが現状である。

また、ZigBee は主にセンサネットワークと組み合わせて使用され、測位技術を身近に利用できる環境を提供している。その一例が Location Engine で、アドホックなマルチホップネットワークにおける RSS 方式の測位を実現している。

第 3 章

Location Engine の課題

ZigBee を用いた測位システムとして Location Engine は注目されているが，いくつかの課題がある．本章では，Location Engine を使用した既存研究を挙げ，ZigBee ノードの電波強度に関する測定について述べる．

3.1 既存研究

本節では，Location Engine を用いた既存研究についてまとめる．

3.1.1 Location Engine を用いたヒトの移動検出に関する研究

仰木ら [16] は，Location Engine の測位精度と時間分解能精度を明らかにし，屋内環境下においてヒトが運動する様子を計測するため，Location Engine を用いて屋内移動履歴の計測を試みた．慶應義塾大学湘南藤沢キャンパスの体育館で行われた実験の結果，電波状況によって大きく測位精度が異なり，数 m 単位で誤差が発生したり，環境内の Blind Node 数に欠落が生じたりすることが確認された．測位誤差は Blind Node が静止しているときにも発生することから，Location Engine を用いた移動体測位は現状では困難であると確認された．

3.1.2 移動する ZigBee センサーノードの位置推定アルゴリズム

米村ら [17] は、Location Engine を用いた移動体測位精度の改善案を提案し実装した。静止時の測位では、測位計算を繰り返すことで位置の測定結果は収束する。一方、移動時の測位では同じ位置で計算を繰り返すことができないので、外れ値の影響を受けて誤差が大きくなる。従来の Location Engine は、位置導出の解法として線型近似した方程式を用いている。そこで、線形近似方程式ではなく、測位の度により正確な非線型方程式を解いて位置を求めるように改善した。実装物を用いた測位実験では、従来の Location Engine と同じ程度の測位精度を実現したが、移動時の測位誤差は改善できなかった。

3.1.3 ZigBee Location Profile における移動センサノードの省電力化プロトコル

小澤ら [18] は、Location Engine の長時間使用のためのプロトコルを実装し、消費電力化の手法を提案した。この研究では、測位の合間のスリープ (ショートスリープ) とデバイスが PAN 内にいるかを検知するためのスリープ (ロングスリープ) を組み合わせた、二段階スリーププロトコルを実装した。シミュレーションと実験の結果、消費電力を 6 分の 1 程度に抑えることが可能となり、1 ヶ月程度の実証実験にも耐えうることが示された。

3.2 ZigBee ノードの通信距離と RSSI の関係

本節では、Location Engine を用いて行った、電波強度の測定実験について述べる。

3.2.1 目的

Location Engine では、最初に測位計算で用いるパラメータを手動で設定する必要がある。このパラメータは、Location Engine を使用する電波伝搬環境によって異なり、例えば屋外と屋内では違うパラメータを設定する必要がある。Location Engine の利用形態は、屋外のみや屋内のみだけではなく、屋内外を通して利用することができる。そこで、Reference Node の配置位置によって、距離と RSSI の関係がどのように変化するかを調査した。

3.2.2 測定概要

測定の手順は以下の通りである。

1. Reference Node を中心とした半径 0.5m の距離に Blind Node を配置する。
2. Blind Node が受信する XY-RSSI Response の RSSI データを調べる。
3. 同一円周上で Blind Node の位置を変えて複数回 RSSI データの測定を行う。
4. 円の半径を 0.5m ずつ、半径 5m まで伸ばして同様の測定を繰り返す。

測定時のノードの配置は、図 3.1 の通りである。以上の測定を、実験室内で Reference Node の位置を変えて 4 箇所で行った。実験の様子を図 3.2 に示す。

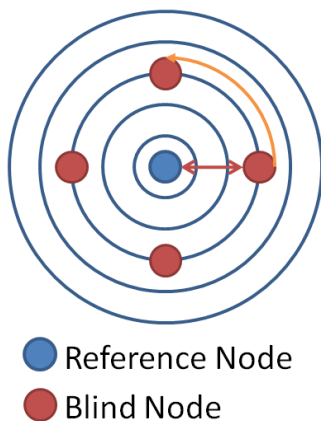


図 3.1 RSSI 測定実験

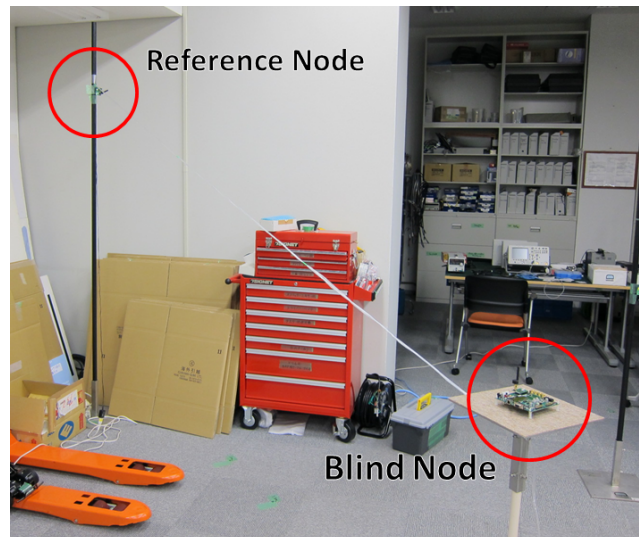


図 3.2 RSSI 測定実験の様子

3.2.3 結果

信号の伝搬距離と RSSI の関係を示したグラフは図 3.3 のようになった。

このグラフをもとに、Reference Node 毎に最適なパラメータを導出した。パラメータ A は、半径 1m における RSSI 値とした。また、パラメータ N は、距離と RSSI の関係から、対数近似を用いて導出した。導出された最適なパラメータは、表 3.1 の通りである。

この結果より、同じ実験室内という環境においても、Reference Node の配置によって最適なパラメータが異なることがわかった。

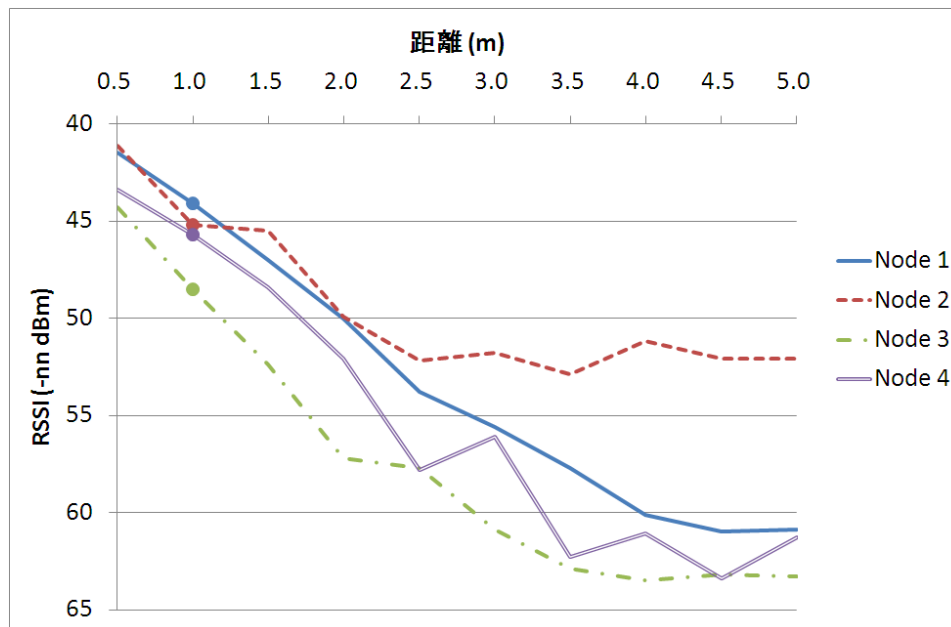


図 3.3 RSSI と距離の関係

表 3.1 Reference Node 毎の最適パラメータ

	A	N
Reference Node 1	44	2.250
Reference Node 2	45	1.250
Reference Node 3	48	2.125
Reference Node 4	45	2.250

3.3 Reference Node の個数と測位精度

本節では、測位計算に用いる Reference Node の個数と測位精度の関係について述べる。

3.3.1 目的

Location Engine は、少なくとも 3 つ以上の Reference Node の存在が検知できれば利用可能である。一般的に、Reference Node の個数が多い程、測位精度が良いと言われている。そこで、測位計算に用いる Reference Node の個数と測位精度の関係を調べた。

3.3.2 実験概要

実験の手順は以下の通りである。

1. 9 個の Reference Node を図 3.4, 3.5 のように配置する。
2. Blind Node を既定の位置に配置する。
3. Location Engine を用いて測位を行い、測位結果の位置座標を得る。
4. Blind Node の位置を変えて、繰り返し測定を行う。

このような測定を、図 3.5 に示した実験室内の 25 箇所でそれぞれ行った。また、測位計算に利用する Reference Node の個数の上限を 3~9 に変更し、それぞれ測位を行った。Location Engine の領域内に複数の Reference Node が存在する場合、測位計算に利用する Reference Node は、RSSI 値が大きいものから順に選択される。実験の様子は図 3.6 の通りである。

その後、実際に Blind Node を配置した位置座標と、Location Engine が導出した位置座標を比較し、距離誤差を計算した。実際の位置座標を $\mathbf{x}$ 、Location Engine が導出した位置座標を $\mathbf{x}_l$ とすると、距離誤差 e_l は、

$$e_l = \|\mathbf{x} - \mathbf{x}_l\|. \quad (3.1)$$

以上の計算より、Blind Node を配置した 25 箇所における、Reference Node の個数を変動させた測位の距離誤差を求めた。

3.3.3 結果

測位計算に用いる Reference Node の個数に応じた測位誤差の分布を図 3.7、その箱ひげ図を図 3.8 に示す。グラフより、使用する Reference Node が少ない場合と比べて、

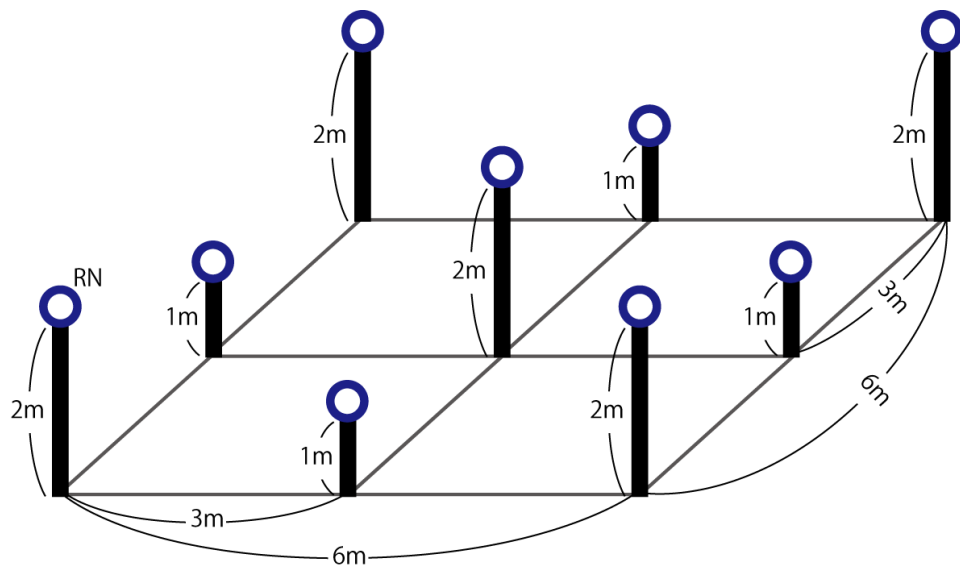


図 3.4 Reference Node の配置図

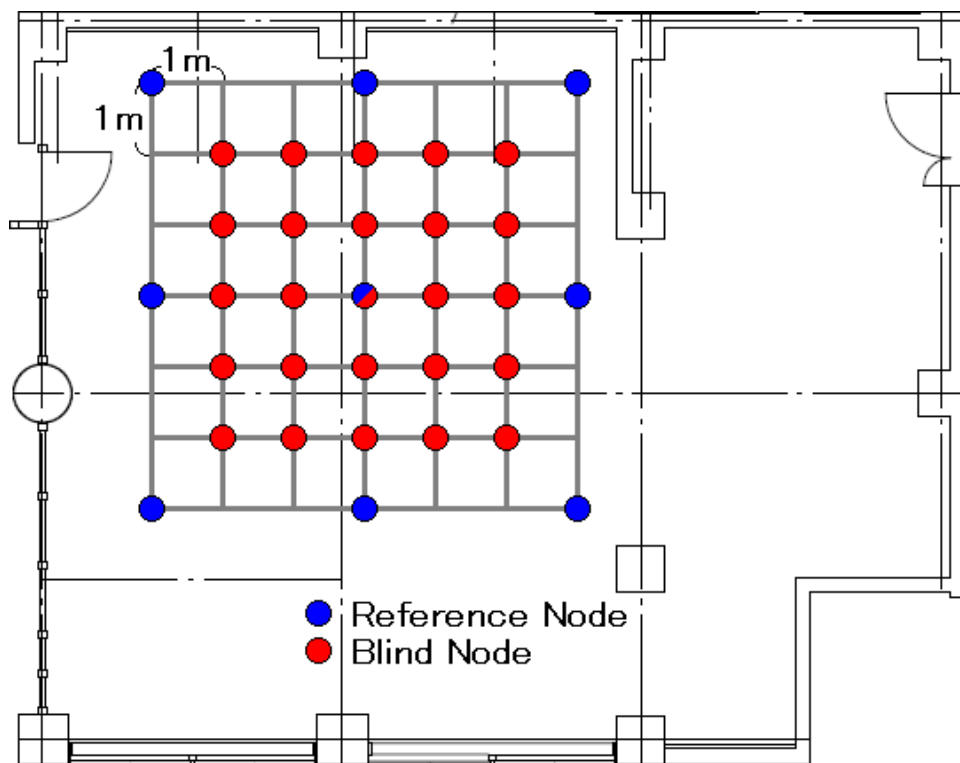


図 3.5 実験室内における各ノードの配置図

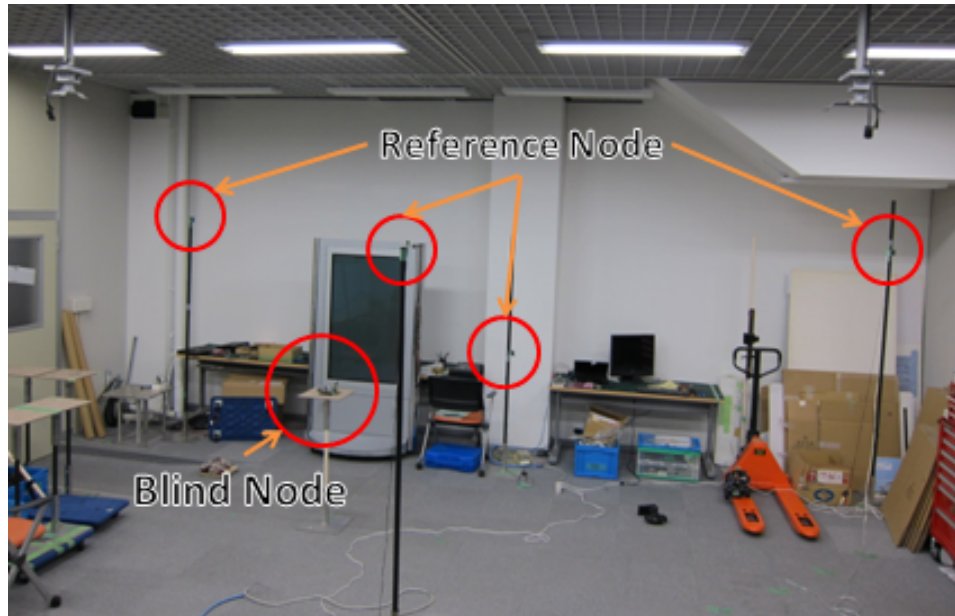


図 3.6 測位実験の様子

Reference Node の個数が増えると測位誤差とそのバラつきは小さくなった。しかし、個数が増えるにつれて測位誤差分布の変化量が小さくなっていった。よって、Reference Node の個数を増やす程、精度向上の変化量は小さくなることがわかる (図 3.9, 3.10)。本実験の環境では、6 個の Reference Node を使用して測位することが妥当である。

3.4 まとめ

Location Engine を用いた既存研究について述べた。既存研究では、移動時の測位精度が好ましくないことや、低消費電力の為にプロトコルの提案など、Location Engine の課題が浮き彫りとなった。

また、RSSI 測定実験の結果、同じ環境内でも Reference Node の位置によって最適なパラメータが異なることがわかった。その結果から、広範囲において Location Engine を使用する場合、最適なパラメータの設定が困難であると考えられる。

そして、Reference Node の個数は多い方が測位精度が良いことが示された。一方、Reference Node の個数が増えるにつれて、測位精度が収束していくことが確認された。

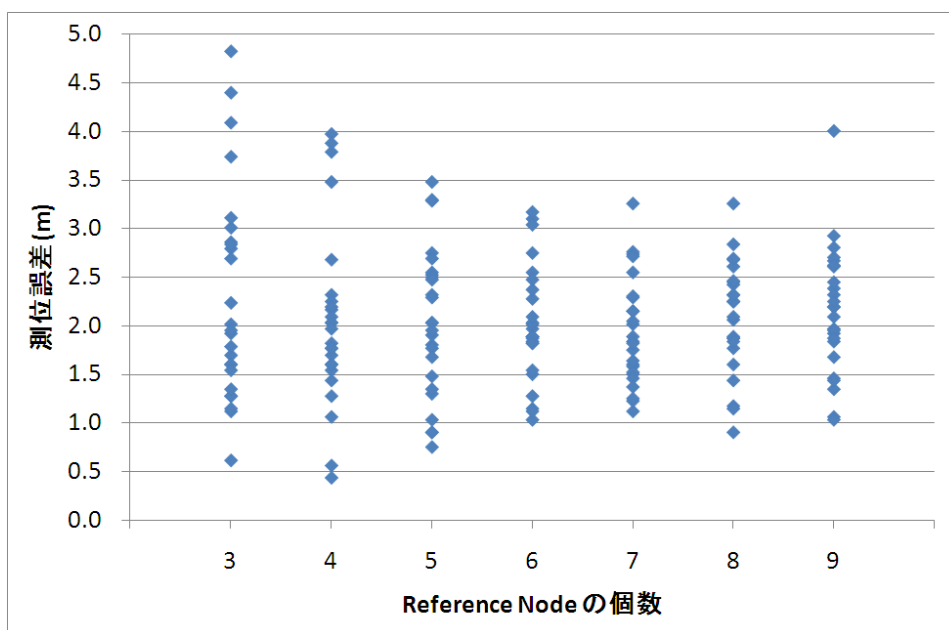


図 3.7 Reference Node の個数毎の測位誤差分布図

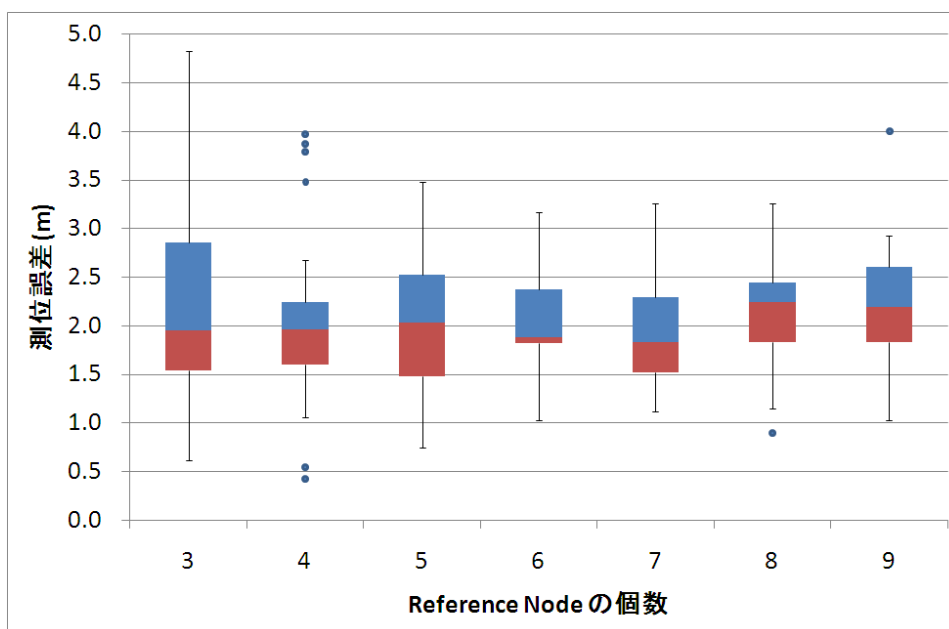
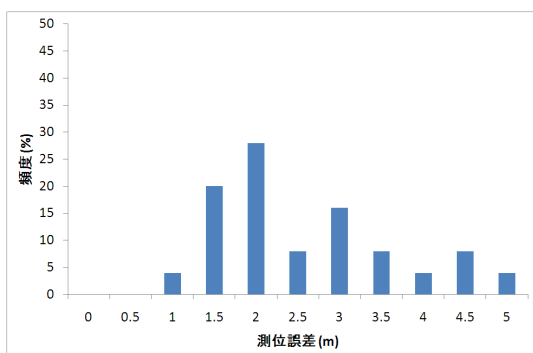
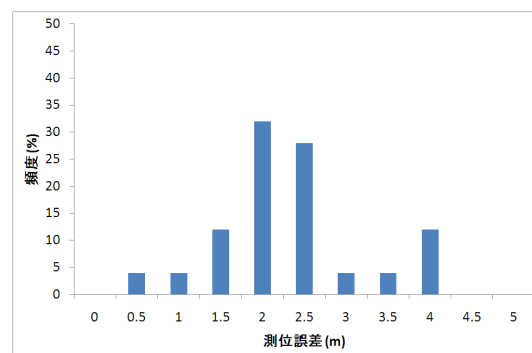


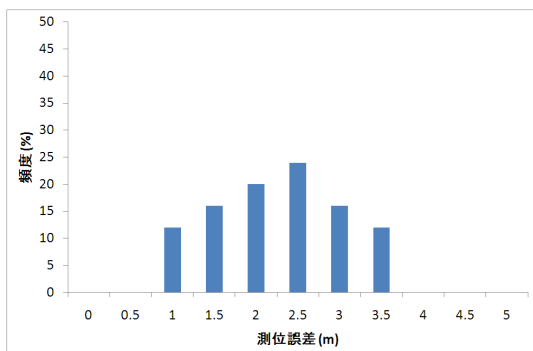
図 3.8 Reference Node の個数毎の測位誤差を示す箱ひげ図



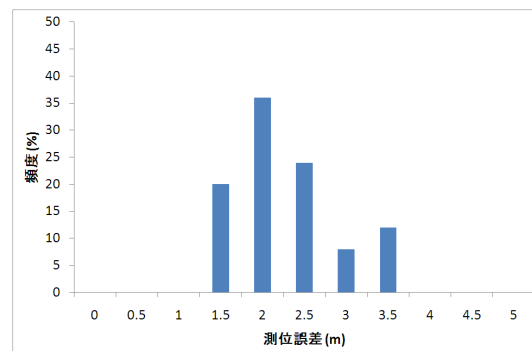
(a) Reference Node 3 個



(b) Reference Node 4 個

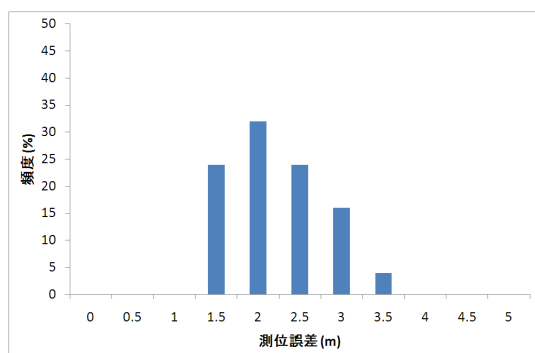


(c) Reference Node 5 個

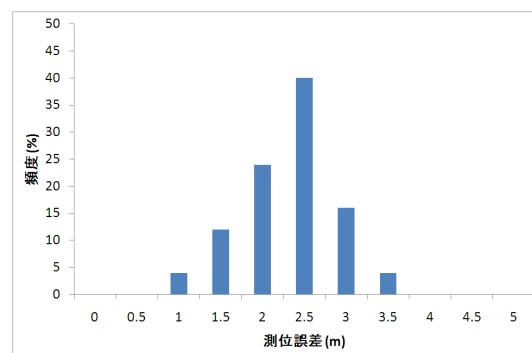


(d) Reference Node 6 個

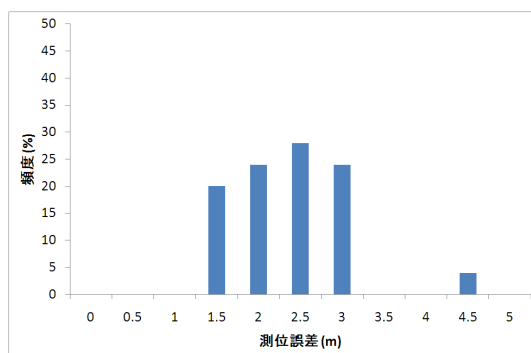
図 3.9 Reference Node の個数毎の測位誤差ヒストグラム (1)



(a) Reference Node 7 個



(b) Reference Node 8 個



(c) Reference Node 9 個

図 3.10 Reference Node の個数毎の測位誤差ヒストグラム (2)

第4章

Location Engine 改善へのアプローチ

従来の Location Engine には、様々な改善の余地がある。本章では、Location Engine の改善に対するアプローチを示す。

4.1 距離導出の計算

本節では、Reference Node と Blind Node の間の距離の計算に着目する。

4.1.1 ソフトウェア実装

Location Engine は CC2431 のハードウェアで実装されている。CC2431 では、MEA-SPARM レジスタに RSSI 値を書き込むと、LOCX/LOCY レジスタから位置座標が読み込める仕組みとなっている。CC2431 は、パラメータ A と N に応じた、RSSI と距離の関係を示す表を保持しているため、ソフトウェア上の計算無しで測位ができる。この位置導出は CC2431 特有のもので、計算過程がブラックボックスになっている。このため、Location Engine は他のデバイスへの移植が困難である。

4.1.2 距離計算の線形近似

Location Engine において、RSSI と距離の関係式は、パラメータによって調整される。距離を RSSI の関数として表すと、

$$d = 10^{-\frac{A+RSSI}{10N}}. \quad (4.1)$$

よって、RSSI から距離を導出するには、指数・対数計算を行わなければならない。しかし、ソフトウェアとして実装する場合、指数・対数計算のような高級関数を使用してしまうと、ファイルサイズが大きくなってしまう。このため、ZigBee チップのようなスタックメモリの小さいものに実装するには難しい要件となる。

そこで本論文では、距離計算は線形近似式を用いて行うことを提案する。第 3 章の 3 節で行った測位実験で得たデータをもとに、距離と RSSI の関係を、線形近似と非線形近似(指数近似)の 2 つの形で求めた(図 4.1)。その結果、線形近似式は

$$d = 0.1122 \cdot RSSI - 2.0175. \quad (4.2)$$

非線形近似式は、

$$d = 0.5855e^{0.0343 \cdot RSSI}. \quad (4.3)$$

そして、求めた 2 つの近似式にそれぞれ、実測実験で得た RSSI 値を代入し直し、距離を再計算した。そして、実際の距離と近似式で再計算して求めた距離の誤差を求めた。線形近似と非線形近似の、再計算における誤差の比較は表 4.1 の通りである。

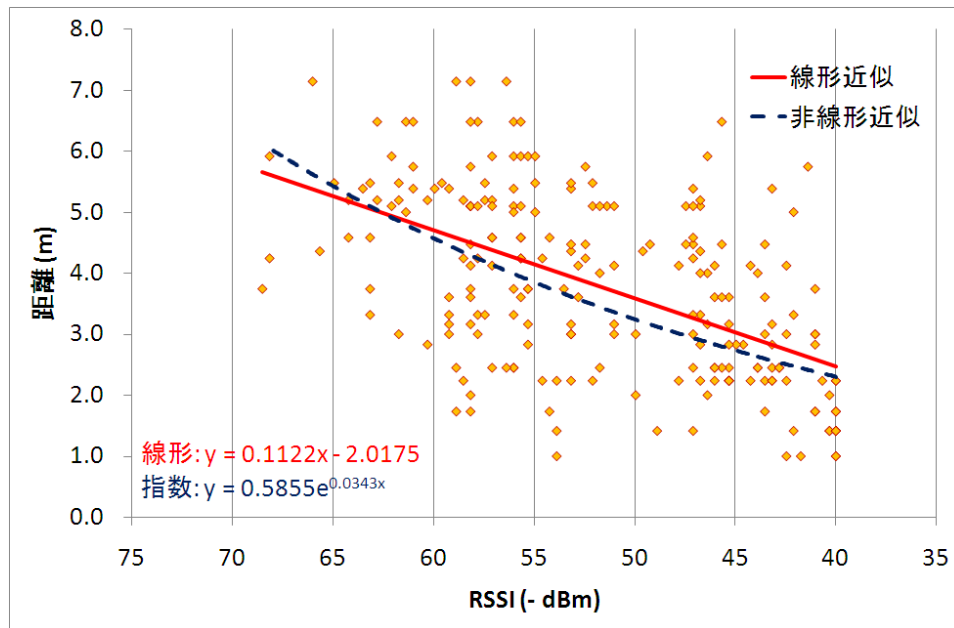


図 4.1 距離と RSSI の関係と近似式

表 4.1 線形近似と非線形近似の測位誤差 (m)

	線形近似	非線形近似
平均値	1.0	1.1
標準偏差	0.7	0.8
最大値	3.4	3.7
最小値	0.0	0.0

RSSI から距離を求める再計算の結果、線形近似と非線形近似では大きな差異は認められなかった。よって、測位計算において線形近似式を用いることは有効である。線形近似式を用いた実装では、使用メモリ領域を小さく抑えることができるので、ZigBee チップへの実装が容易になる。

4.1.3 測位計算アルゴリズム

線形近似式を用いた Blind Node の位置座標導出の過程を示す。Reference Node の配置で高さを考慮するため、求める Blind Node の位置座標を、

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (4.4)$$

とし、3次元測位を前提として考える。また、ソフトウェアのファイルサイズを小さく抑えるため、指数対数や累乗根のような高機能な演算を使用しないことにする。

距離算出の線形近似式における傾きと切片のパラメータを a, b とすると、 i 番目の Reference Node より受信した $RSSI_i$ から算出した距離 l_i は、

$$l_i = a \cdot RSSI_i + b. \quad (4.5)$$

そして、 i 番目の Reference Node($\mathbf{x}_i$) と Blind Node($\mathbf{x}$) の実距離の二乗と算出距離 l_i の二乗の差分 e_i は、

$$e_i = \|\mathbf{x} - \mathbf{x}_i\|^2 - l_i^2. \quad (4.6)$$

e_i の二乗和を I とおくと、

$$I = \sum_i^N e_i^2 \quad (4.7)$$

最適解 $\mathbf{x}$ の算出には、最小二乗法を用いる。よって、この I が最小になるように、 $\mathbf{x}$ を求める。

I が最小となる条件は、

$$\begin{aligned} f(\mathbf{x}) &= \begin{pmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{pmatrix} I \\ &= \mathbf{0} \end{aligned} \quad (4.8)$$

である。

ここからニュートン法を用いて、 $f(\mathbf{x}) = \mathbf{0}$ の最適解 $\mathbf{x}$ を求める。 $\mathbf{x}$ の初期値を $\mathbf{x}_0$ とし、 $f(\mathbf{x})$ を一次近似すると、

$$\begin{aligned} f(\mathbf{x}) &\approx f(\mathbf{x}_0) + \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}_0)(\mathbf{x} - \mathbf{x}_0) \\ &= 0. \end{aligned} \quad (4.9)$$

$$\therefore \mathbf{x} = \mathbf{x}_0 - \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}_0)^{-1} f(\mathbf{x}_0) \quad (4.10)$$

ここで、

$$\mathbf{r} = \frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}_0)^{-1} f(\mathbf{x}_0) \quad (4.11)$$

とおくと、

$$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}_0) \cdot \mathbf{r} = f(\mathbf{x}_0). \quad (4.12)$$

$\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}_0)$ は 3 次対称行列であるから、 LDL^T 分解が可能である。 LDL^T 分解後の方程式を、前進・後退代入法で解いて $\mathbf{r}$ を求め、

$$\mathbf{x} = \mathbf{x}_0 - \mathbf{r} \quad (4.13)$$

に代入する。 $\mathbf{r}$ が十分に小さければ、 $\mathbf{x}$ の更新を止める。 $\mathbf{x}$ を更新した後は、再び式 4.6 から計算する。

これを繰り返して、最適解 $\mathbf{x}$ を求める。

4.2 パラメータの自動校正

本節では、測位計算に用いるパラメータに着目する。

4.2.1 Reference Node 毎の最適パラメータ

従来の Location Engine では、RSSI から距離を計算するときにパラメータ A と N を用いる。このパラメータは、Location Engine の使用環境に応じてユーザが手動で設定しなければならない。最適なパラメータは、第 3 章で述べたような実測実験によってしか得られない。このような設定方法は煩雑である。そこで本論文では、パラメータを自動校正で設定する手法を提案する。

RSSI 値測定実験の結果より、測位計算に用いるパラメータは、Reference Node の配置によって最適値が異なる。周りに障害物がない Reference Node は RSSI の減衰量が小さいが、周りを壁に囲まれていたり障害物の多い空間にある Reference Node では RSSI の減衰量が大きくなる。従来の Location Engine では、測位計算に用いるパラメータを Blind Node が保持しているため、個々の Reference Node 周辺の環境が大きく異なるときには、適切なパラメータの設定が困難であった。そこで、Reference Node 毎にパラメータを導出し、Blind Node に送信するプロトコルを提案する。

4.2.2 システム構成

Reference Node が距離と RSSI の関係を表すパラメータを自動校正するには、距離が既知である他のノードとの通信が必要となる。

そこで、新たに Reference Node が RSSI Blast のブロードキャストメッセージを送信し、そのメッセージを他の Reference Node が受信して RSSI 値を得る。複数の Reference Node からブロードキャストメッセージを受信し、集めたデータから距離と RSSI の関係を求める。距離と RSSI の関係は線形近似式で表すので、得られた近似式の傾きと切片をパラメータとする。導出したパラメータは、XY-RSSI Response のパッケージに組み込んで、自身の座標と RSSI 値とともに Blind Node へ送信される。XY-RSSI Response のパッケージは、従来の Location Engine では X 座標・Y 座標と RSSI を含む (図 4.2)。本手法では、さらに Z 座標と傾きパラメータ A、切片パラメータ B を加えたものとなる (図 4.3)。Reference Node のブロードキャストメッセージを含んだメッセージシーケンスを図 4.4 に示す。

...	X 座標	Y 座標	RSSI	...
-----	------	------	------	-----

図 4.2 従来版 Location Engine の XY-RSSI Response パケット

...	X 座標	Y 座標	Z 座標	パラメータ A	パラメータ B	RSSI	...
-----	------	------	------	---------	---------	------	-----

図 4.3 改良版測位システムの XY-RSSI Response パケット

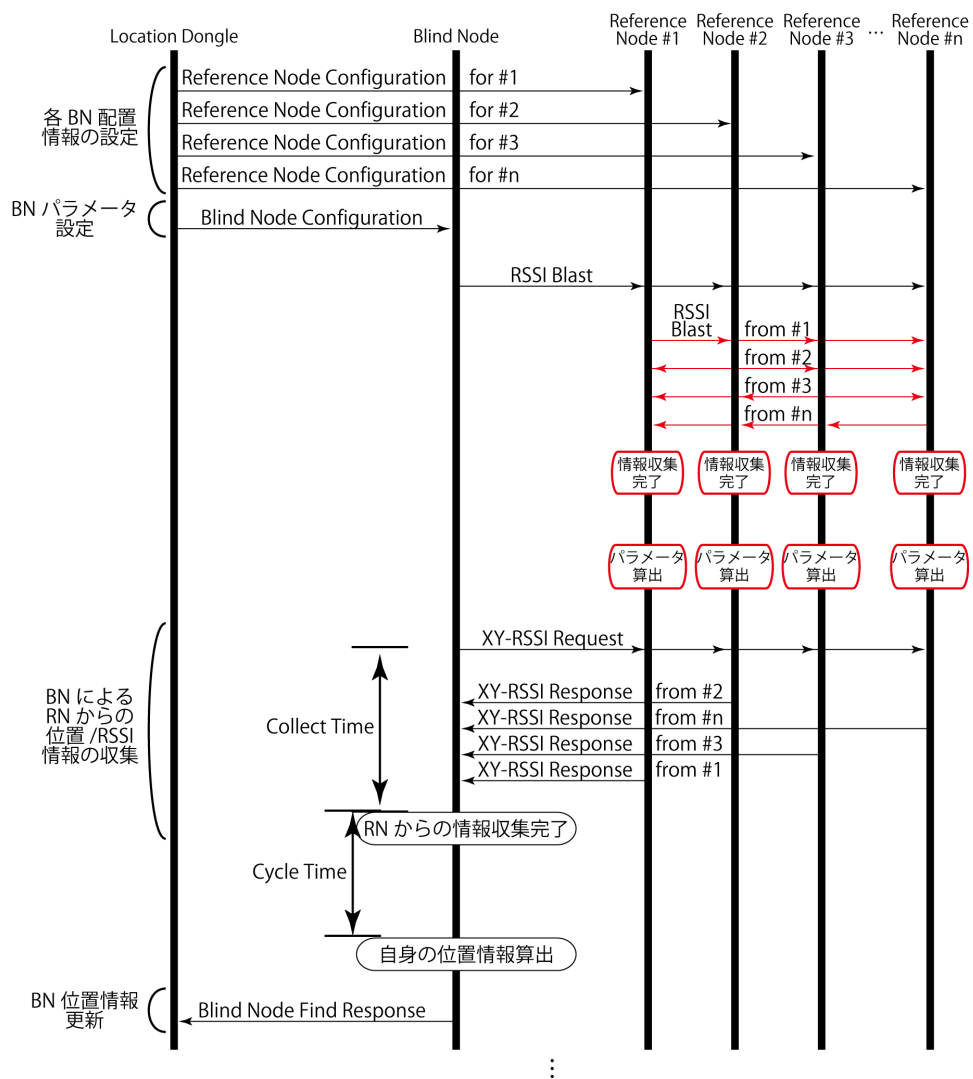


図 4.4 改良版測位システムのメッセージシーケンス

4.3 まとめ

本章では、Location Engine の改善案について述べた。Location Engine をソフトウェアベースで実装するにあたり、距離計算を非線形計算ではなく線形近似を用いて行うことを提案した。距離と RSSI の関係を線形近似と非線形近似で表して距離計算の精度を比較した結果、2つの近似式を比べても誤差はほとんど変わらなかった。距離計算を線形近似で実装することで、ファイルサイズを小さく抑えられる。

また、Location Engine では使用環境に応じてパラメータを手動で設定しなければならない。さらに、Reference Node は配置場所によって信号の伝搬環境が異なることから、最適なパラメータの設定が困難であった。パラメータ設定の簡易化と測位精度の改善のため、Reference Node 毎にパラメータを自動校正して、Blind Node に送信するプロトコルを提案した。

第 5 章

改良版測位システムの評価

本章では，Location Engine をもとに提案したソフトウェア測位システムの特徴をまとめ，その評価を示す．

5.1 測位システムのソフトウェア実装

Location Engine のプログラムを利用して，改良版測位システムをソフトウェアで実装することを提案する．開発環境は表 5.1 の通りである．

表 5.1 開発環境

実装マシン	ThinkPad T400
OS	Microsoft Windows XP SP3
開発言語	C
IDE	IAR Embedded Workbench
開発ボード	CC2530EM
スタックメモリサイズ	250KB

従来はハードウェアで実装されていた距離導出から位置算出までの過程を，ソフトウェアベースで実装する．距離と RSSI の関係より，正確な距離計算を行うには指数・対数計算を用いる必要がある．しかし，スタックメモリサイズの小さい ZigBee チップに実装する際，指数・対数計算のような高級関数の使用は要件的に厳しい．そこで，指数・対数計

算を用いず，線形計算のみで位置を計算することでファイルサイズを小さく抑えることができる．従来の Location Engine・線形測位・非線形測位を実装したファイルサイズの比較を表 5.2 に示す．

表 5.2 Blind Node ファイルサイズの比較

	距離計算実装形式	ファイルサイズ
Location Engine	レジスタ演算	204,606 byte
測位システム A	ソフトウェアの線形近似計算	216,381 byte
測位システム B	ソフトウェアの非線形計算	219,188 byte

従来の Location Engine は，測位計算がハードウェア実装であるため，ソフトウェアベースで実装されている測位システムと比べてプログラムのファイルサイズ自体は小さい．また，線形測位と非線形測位のファイルサイズを比較すると，線形測位の方がファイルサイズを小さく抑えることができる．これは，非線形測位では，指数・対数計算といった高級関数の参照の影響であると考えられる．

5.2 自動校正測位のシミュレーション

本節では，本論文で提案する自動校正手法を評価するためのシミュレーションについて述べる．

5.2.1 ACPS の実装

本論文では，測位計算に用いるパラメータを自動校正する測位システム，ACPS(Automatic Calibration Positioning System) を提案する．ACPS は，Reference Node 毎に距離と RSSI の関係を示すパラメータを自動校正し，Blind Node に送信して距離計算に用いる測位システムである．信号の減衰を空間で考慮するため，3次元で測位を行う．ACPS では，パラメータは自動的に設定されるため，Reference Node の座標の設定さえすれば測位が可能になる．ASPC を含む各測位システムの特徴を，表 5.3 にまとめる．

表 5.3 測位システムの特徴

	Location Engine	2D 測位	3D 測位	ACPS
Reference Node 配置位置設定	必要	必要	必要	必要
Blind Node パラメータ設定	必要	必要	必要	不要
測位実装形式	レジスタ演算	ソフトウェア	ソフトウェア	ソフトウェア
距離計算	指数計算	線形近似計算	線形近似計算	線形近似計算
測位解法	線形近似方程式	ニュートン法	ニュートン法	ニュートン法
測位座標	2 次元	2 次元	3 次元	3 次元
パラメータ自動校正	無し	無し	無し	有り

5.2.2 測位シミュレーション実験

測位精度を比較するため、測位シミュレーションを行った。比較する測位システムは、Location Engine とソフトウェア実装の 2D 測位の 2 つの 2 次元測位、ソフトウェア実装の 3D 測位と ACPS の 2 つの 3 次元測位である。シミュレーションに用いるデータは、第 3 章 3 節で行った測位実験の手法で得た。

実験の手順は以下の通りである。

1. 9 個の Reference Node を図 5.1, 5.2 のように配置する。
2. Blind Node を既定の位置に配置する。
3. Sensor Network Analyzer を用いて、XY-RSSI Response のパケットデータを得る。
4. X 座標, Y 座標, Z 座標, パラメータ A, パラメータ B, RSSI 値をパケットデータから抽出する。
5. Blind Node の位置を変えて、繰り返し測定を行う。

このような測定を、図 5.2 の 25 箇所において行った。25 箇所の測位で得られたパケットデータから、各測位システムのモデルを用いて位置測定のシミュレーションを行った。測位に用いる Reference Node の個数は、第 3 章 3 節で求めた最適値である 6 個とした。

5.2.3 測位誤差の比較

実際に Blind Node が配置された位置座標と測位で得られた位置座標を比較し、誤差を調べた。シミュレーションによって得られた位置座標データは、2 次元測位の場合は X 座標・Y 座標、3 次元測位ではそれらに加えて Z 座標である。

線形近似と非線形近似の測位シミュレーション誤差比較を図 5.3, 図 5.4 と表 5.4 に示す。

2 次元測位では、ソフトウェア実装の 2D 測位は Location Engine と比べて誤差が大きくなった。Location Engine では、距離計算をレジスタに依存した非線形計算で行っている。一方、2D 測位では、距離計算を線形近似したため、誤差が大きくなっていた。2D 測位における誤差は、ニュートン法の計算で収束させても改善されなかった。

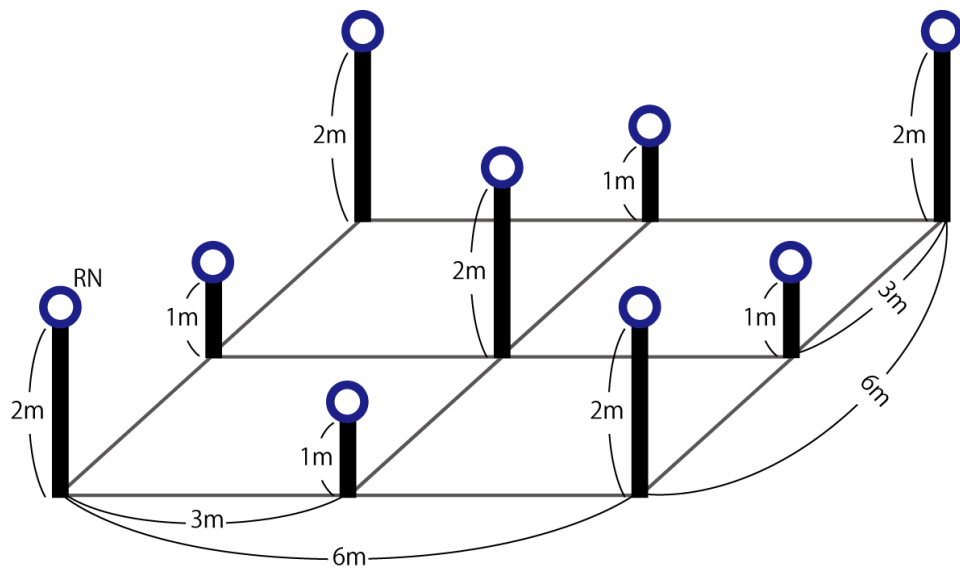


図 5.1 Reference Node の配置図 (再掲)

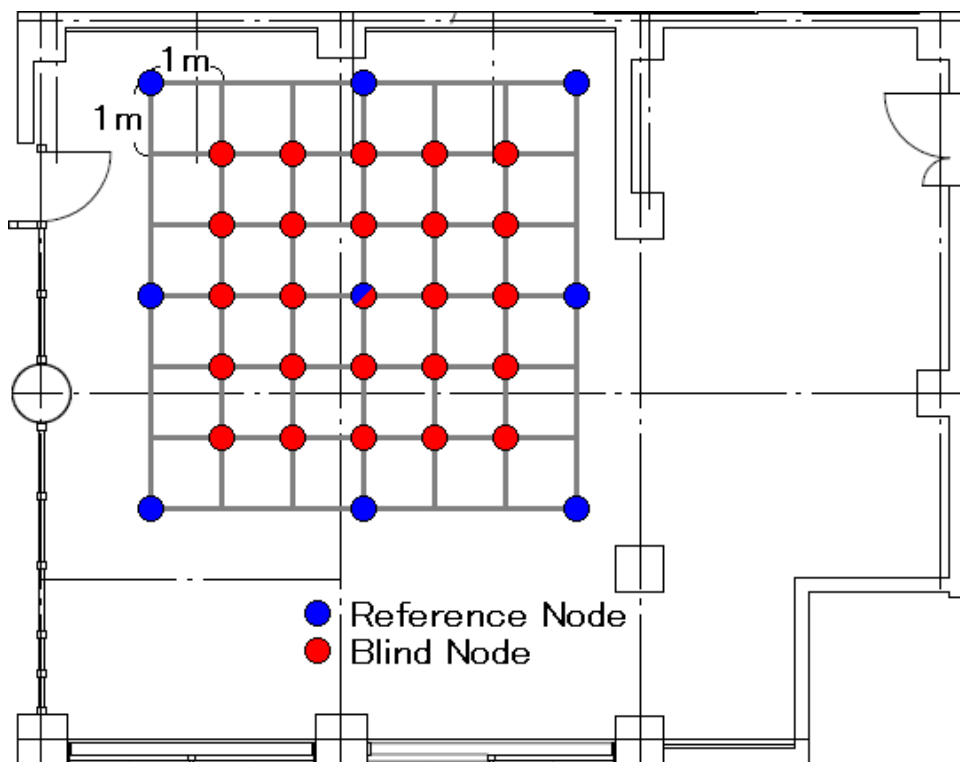
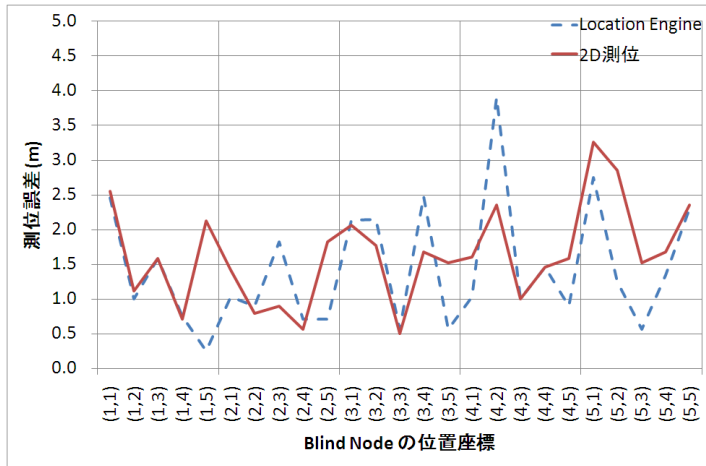
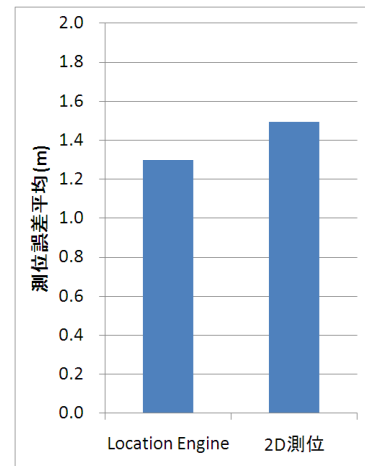


図 5.2 実験室内における各ノードの配置図 (再掲)

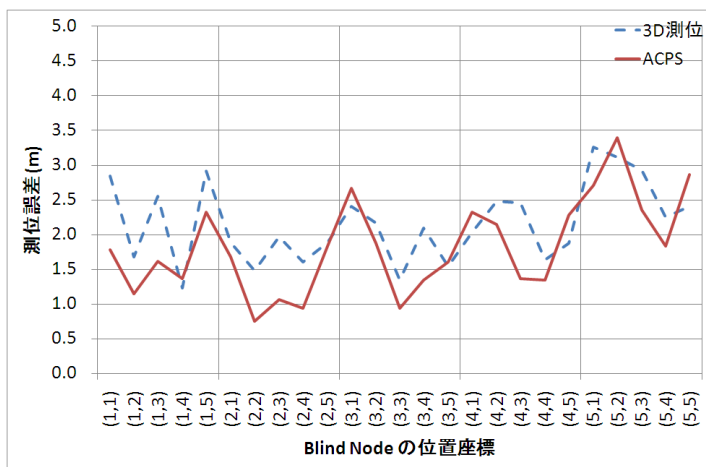


(a) 測位誤差

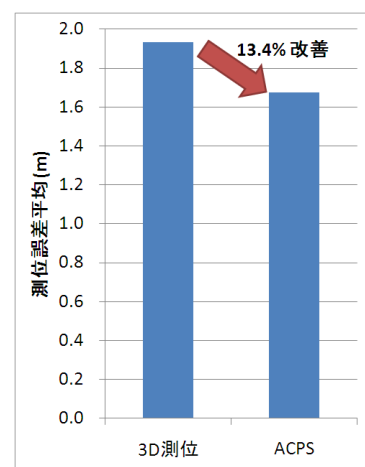


(b) 測位誤差の平均

図 5.3 2次元測位の誤差



(a) 測位誤差



(b) 測位誤差の平均

図 5.4 3次元測位の誤差

表 5.4 測位誤差の比較 (m)

	Location Engine	2D 測位	3D 測位	ACPS
平均値	1.3	1.5	1.9	1.7
標準偏差	1.1	1.0	1.1	1.0
最大値	3.9	3.3	3.3	3.4
最小値	0.3	0.5	1.2	0.8

3次元測位では、自動較正無しの3D測位と比べてACPSは測位誤差が小さくなり、測位精度は13.4%改善された。3D測位では、あらかじめ手動で設定した1組のパラメータをもとに全ての測位計算を行っている。一方、ACPSでは、Reference Nodeが自身の環境に即してパラメータを導出するため、各Reference Nodeが送信したXY-RSSI Responseの一つ一つに対応したパラメータを用いて測位を行っている。このため、測位誤差が小さく抑えられ、測位精度が改善されることが示された。

5.3 まとめ

本章では、Location Engineをソフトウェア実装して改良したACPSの評価を行った。線形近似測位におけるファイルサイズは、非線形計算測位のそれと比べて小さく抑えられた。よって、測位システムにおける距離計算が簡易化され、さらに他のZigBeeチップへの実装が容易になった。ソフトウェアベースの測位システムでは、3次元測位が可能となる。

また、パラメータを自動較正するACPSを提案した。ACPSによって、Reference Nodeの座標の設定さえすれば測位が可能になった。さらに、ACPSを利用することで、3次元測位の精度が13.4%改善されることを示した。

第 6 章

結論と今後の展望

本論文の結論である．

6.1 結論

本論文では，ZigBee を用いた 3 次元屋内測位システムを提案し，その性能を評価した結果を報告した．ZigBee を用いた屋内測位システムは，一般的に RSSI を元にした三角測量を測定原理としている．Reference Node と Blind Node の間の距離と通信パケットの RSSI の関係は対数関係で表されることが多いが，たとえ対数関係で表したとしても必ずしも測位精度は改善されない．逆に対数関係で表すことは，ZigBee チップ内で非線形計算を行わねばならない上，さらに特定のハードウェアでないと測位ができないことにつながってしまう．RSSI と距離の関係を線形近似したとしても，三角測量に関係する参照点である Reference Node の個数を増やすことによって，測位精度を大幅に劣化させることなく測位計算を簡易化できる．この線形近似により，250kB のスタックメモリを有する市販 ZigBee チップでも，ソフトウェアだけで 3 次元の非線形測位計算を行うことができる．また，RSSI を用いた測位では，測定する場所に応じて RSSI と距離の関係が変化してしまうことが，測位誤差の大きな要因である．これを解決するために，Reference Node 毎に周辺環境における RSSI と距離の関係をあらかじめ測定しておき，Reference Node に通知するプロトコル，ACPS(Automatic Calibration Positioning System) を考案した．ACPS を用いた場合には，用いないと比べて 3 次元測定で最大 13.4% の測定精度向上が実現できる．

6.2 今後の展望

今後の展望について 2 点述べる．

6.2.1 移動時の測位

Location Engine や ACPS では，位置座標の算出において最終的に前回の位置座標を参照して，フィルタリングを行っている．よって静止時の測位では，測位を繰り返すことで位置座標が収束する．しかし，移動時には前回の位置座標は現在の位置と異なるため，位置座標が収束せず測位精度が悪い．移動時の測位精度改善について議論する必要がある．

6.2.2 RSS 方式以外の測位

RSS 方式の測位は，信号の伝搬環境に大きく影響を受ける．伝搬環境は人の移動や障害物によって変動するため，実空間における測位精度向上には限界がある．電波を用いた測位では RSS 方式の測位のみではなく，ToA/TDoA 方式や AoA 方式，または複数の方式を組み合わせる必要がある．

付録 A

LDL^T 分解と前進・後退代入法

本付録では、測位システムのソフトウェア (ACPS) 実装において、測位計算で用いる LDL^T 分解と前進・後退代入法について述べる。

A.1 LDL^T 分解

$\mathbf{A}$ を 3 次の対称行列, $\mathbf{L}$ を対角成分が 1 である 3 次の下三角行列, $\mathbf{D}$ を 3 次の対角行列とする。

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{21} & a_{31} \\ a_{21} & a_{22} & a_{32} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}, \mathbf{L} = \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix}, \mathbf{D} = \begin{pmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{pmatrix}$$

このとき, $\mathbf{A}$ を,

$$\mathbf{A} = \mathbf{LDL}^T$$

の形に分解することができる。

$$\begin{aligned}
A &= LDL^T \\
&= \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{pmatrix} \begin{pmatrix} 1 & l_{21} & l_{31} \\ 0 & 1 & l_{32} \\ 0 & 0 & 1 \end{pmatrix} \\
&= \begin{pmatrix} d_{11} & d_{11}l_{21} & d_{11}l_{31} \\ d_{11}l_{21} & d_{11}l_{21}^2 + d_{22} & d_{11}l_{21}l_{31} + d_{22}l_{32} \\ d_{11}l_{31} & d_{11}l_{21}l_{31} + d_{22}l_{32} & d_{11}l_{31}^2 + d_{22}l_{32}^2 + d_{33} \end{pmatrix}
\end{aligned} \tag{A.1}$$

よって、以下の順に計算し、

$$d_{11} = a_{11}, \tag{A.2}$$

$$l_{21} = a_{21}/d_{11}, \tag{A.3}$$

$$l_{31} = a_{31}/d_{11}, \tag{A.4}$$

$$d_{22} = a_{22} - d_{11}l_{21}^2, \tag{A.5}$$

$$l_{32} = (a_{32} - d_{11}l_{21}l_{31})/d_{22}, \tag{A.6}$$

$$d_{33} = a_{33} - d_{11}l_{31}^2 - d_{22}l_{32}^2. \tag{A.7}$$

以上より、 $A = LDL^T$ が成立することが示された。

A.2 前進・後退代入法

A は 3 次の対称行列、 b は 3 次のベクトル $\left(b = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \right)$ とする。このとき、

$$Ax = b$$

となるような 3 次のベクトル $x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ の解を求める。

A を LDL^T 分解すると、

$$A = LDL^T. \tag{A.8}$$

よって,

$$LDL^T \mathbf{x} = \mathbf{b}. \quad (\text{A.9})$$

ここで,

$$\mathbf{y} = DL^T \mathbf{x} \quad (\text{A.10})$$

となる 3 次のベクトル $\mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$ をおき, $\mathbf{y}$ の解を求める. よって,

$$L\mathbf{y} = \mathbf{b}. \quad (\text{A.11})$$

$$\therefore \begin{pmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \quad (\text{A.12})$$

ここで, 連立方程式を上から順に解くと (前進代入法),

$$y_1 = b_1, \quad (\text{A.13})$$

$$y_2 = b_2 - l_{21}y_1, \quad (\text{A.14})$$

$$y_3 = b_3 - l_{31}y_1 - l_{32}y_2. \quad (\text{A.15})$$

よって, $\mathbf{y}$ の解が求められた. ところで,

$$DL^T \mathbf{x} = \mathbf{y} \quad (\text{A.16})$$

であるから,

$$\begin{aligned} \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} &= \begin{pmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{pmatrix} \begin{pmatrix} 1 & l_{21} & l_{31} \\ 0 & 1 & l_{32} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \\ &= \begin{pmatrix} d_{11} & d_{11}l_{21} & d_{11}l_{31} \\ 0 & d_{22} & d_{22}l_{32} \\ 0 & 0 & d_{33} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}. \end{aligned} \quad (\text{A.17})$$

ここで, 連立方程式を下から順に解くと (後退代入法),

$$x_3 = y_3/d_{33}, \quad (\text{A.18})$$

$$x_2 = (y_2 - d_{22}l_{32}x_3)/d_{22}, \quad (\text{A.19})$$

$$x_1 = (y_1 - d_{11}l_{21}x_2 - d_{11}l_{31}x_3)/d_{11}. \quad (\text{A.20})$$

以上より, $\mathbf{x}$ の解が求められた.

付録 B

測位システムソースコード

本付録では、測位システムのソフトウェア (ACPS) 実装において、Location Engine の Blind Node 用プログラムに含まれる関数 `locationCalculatePosition` を改良したものを載せる。

ソースコード B.1 `locationCalculatePosition`

```

/* hiroishi calculation */
void locationCalculatePosition( LocRefNode_t *ref, LocDevCfg_t *node )
{
    LocRefNode_t *pRef = ref;
    // Rev-B Chip have LocEng Ver 1.0 w/ cap=8, Rev-C have LocEng Ver 2.0 w/
    16.
    const byte stop = ( ( CHVER == 0x01 ) ? LOC_ENGINE_NODE_CAPACITY_REVB :
        LOC_ENGINE_NODE_CAPACITY_REVC );
    byte idx;

    int RefDist[BLINDNODE_MAX_REF_NODES];
    int xvar, yvar, zvar;
    int xtmp, ytmp, ztmp;
    long e;
    long M[6], V[3];
    byte RefCnt = 0;
    byte CalcCnt, i;

    /* B N          u R S S I          R N          u          */
    xvar = pRef->x; yvar = pRef->y; zvar = pRef->z;

    /* RN- B N          Z o          */
    for( idx = 0; idx < stop; idx++ ) {
        if( pRef->rss_i == 0 || pRef->rss_i > 80 ) break;
        RefDist[idx] = ( pRef->param_s*pRef->rss_i + pRef->param_i*32 ) / 64;
        RefCnt++;
        pRef++;
    }

    /* j          [ g @ B N          u          Z o          */
    for( CalcCnt = 0; CalcCnt < 10; CalcCnt++ ) {

```

```

    pRef = ref;
    for( i = 0; i < 6; i++ ) M[i] = 0;
    for( i = 0; i < 3; i++ ) V[i] = 0;

    for( i = 0; i < RefCnt; i++ ) {
        xtmp = pRef->x - xvar;
        ytmp = pRef->y - yvar;
        ztmp = pRef->z - zvar;
        e = xtmp*xtmp + ytmp*ytmp + ztmp*ztmp - RefDist[i]*RefDist[i];

        M[0] += 2*xtmp*xtmp + e;
        M[1] += 2*xtmp*ytmp;
        M[2] += 2*ytmp*ytmp + e;
        M[3] += 2*xtmp*ztmp;
        M[4] += 2*ytmp*ztmp;
        M[5] += 2*ztmp*ztmp + e;
        V[0] -= xtmp*e;
        V[1] -= ytmp*e;
        V[2] -= ztmp*e;
        pRef++;
    }
    Sol(M, V);
    if(V[0]*V[0] + V[1]*V[1] + V[2]*V[2] < 10) break;
    xvar -= V[0];
    yvar -= V[1];
    zvar -= V[2];
}

/* B N      u      X V */
node->x = V[0];
node->y = V[1];
node->z = V[2];
if( CHVER == 0x01 ) node->min = LOCMIN;
else node->min = 0;
}

/*      ^      o */
// "Mx = V"      x
void Sol(long* m, long* v)
{
    float dm0, dm1, dm2, dm3, dm4, dm5;
    float dv0, dv1, dv2;
    byte f = 0;
    dm0 = (float)m[0]; dm1 = (float)m[1]; dm2 = (float)m[2];
    dm3 = (float)m[3]; dm4 = (float)m[4]; dm5 = (float)m[5];
    dv0 = (float)v[0]; dv1 = (float)v[1]; dv2 = (float)v[2];

    // M L D L
    // M = LD(L^T)
    if( dm0 == 0 ) f = 1;
    dm1 = dm1/dm0;
    dm3 = dm3/dm0;
    dm2 = dm2 - dm1*dm1*dm0;
    if( dm2 == 0 ) f = 1;
    dm4 = (dm4 - dm1*dm3*dm0)/dm2;
    dm5 = dm5 - dm0*dm3*dm3 - dm4*dm4*dm2;
    if( dm5 == 0 ) f = 1;

    //
    //      u      X V

```

```

    if( f == 1 ) {
        v[0] = 0; v[1] = 0; v[2] = 0;
        return;
    }

    // O i E @
    //
    //  $D(L^T)x = y$ 
    //  $Ly = V$ 
    // y O i @
    dv1 = dv1 - dm1*dv0;
    dv2 = dv2 - dm3*dv0 - dm4*dv1;
    //  $D(L^T)x = y$ 
    // x @
    dv2 = dv2/dm5;
    dv1 = (dv1 - dm2*dm4*dv2)/dm2;
    dv0 = (dv0 - dm3*dm0*dv2 - dm1*dm0*dv1)/dm0;

    // x l
    v[0] = (long)dv0; v[1] = (long)dv1; v[2] = (long)dv2;
    return;
}

```

謝辞

本論文執筆にあたり，ご指導いただきました慶應義塾大学環境情報学部教授村井純博士，並びに同学部教授徳田英幸博士，同学部准教授楠本博之博士，同学部教授中村修博士，同学部准教授高汐一紀博士，同学部専任講師重近範行博士，同学部准教授 Rodney D. Van Meter III 博士，同学部准教授植原啓介博士，同学部専任講師中澤仁博士，同学部教授武田圭史博士に感謝いたします。

また，研究を進めるにあたり，ご指導とご助言をいただきました慶應義塾大学環境情報学部准教授三次仁博士，並びに同大学大学院政策・メディア研究科特別研究講師羽田久一博士，同研究科特別研究講師稲葉達也博士，同研究科特別研究助教鈴木茂哉氏，同研究科特別研究助教中根雅文氏に感謝いたします。特に，三次仁博士には自身がお忙しい中で本論文執筆の際にご指導ご教鞭いただきました。三次博士無しには本論文の完成はあり得ませんでした。深く感謝いたします。

Auto-ID Lab. Japan において研究生活を共にした慶應義塾大学大学院政策・メディア研究科博士課程神谷尚保氏，並びに同研究科修士課程江村圭吾氏，鈴木詩織氏，山口修平氏，同大学環境情報学部山田真弘氏，宮崎圭太氏，能島良和氏，佐藤友紀氏，横石雄大氏，Doan Hoai Nam 氏，五十嵐祐貴氏，同大学総合政策学部米村茂氏，富田千智氏，同大学商学部齋藤俊氏に感謝いたします。また，研究活動においてお世話になった電気通信大学電気通信学部特任助教川喜多佑介博士，慶應義塾大学大学院メディアデザイン研究科修士課程杉本健一氏，Auto-ID Lab. Japan 研究員金仙麗氏，株式会社日放電子白石雅彦氏に感謝いたします。そして，徳田・村井・楠本・中村・高汐・重近・バンミーター・植原・三次・中澤・武田合同研究プロジェクトの皆様に感謝いたします。

最後に，私生活を支えてくれた家族，そして弓道の修練を共にした慶應義塾大学弓道同好会の友人への感謝を持って，本論文の謝辞といたします。

参考文献

- [1] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle. GNSS のすべて: GPS、グロナス、ガリレオ…。古今書院, 2010. 2.1.1
- [2] KC Ho and YT Chan. Solution and performance analysis of geolocation by TDOA. *Aerospace and Electronic Systems, IEEE Transactions on*, Vol. 29, No. 4, pp. 1311–1322, 2002. 2.1.1
- [3] D. Manandhar, S. Kawaguchi, M. Uchida, M. Ishii, and H. Torimoto. IMES for Mobile Users Social Implementation and Experiments based on Existing Cellular Phones for Seamless Positioning. In *Proc. of the Int. Symposium on GPS/GNSS 2008*, 2008. 2.1.2
- [4] 暦本純一, 塩野崎敦, 末吉隆彦, 味八木崇. PlaceEngine: 実世界集合知に基づく WiFi 位置情報基盤. インターネットコンファレンス, pp. 95–104, 2006. 2.1.3
- [5] 河口信夫. Locky.jp: 無線 LAN を用いた位置推定とその応用. 電子情報通信学会技術研究報告, Vol. 107, pp. 1–40, 2007. 2.1.3
- [6] A. Madhavapeddy and A. Tse. A study of bluetooth propagation using accurate indoor location mapping. *UbiComp 2005: Ubiquitous Computing*, pp. 105–122, 2005. 2.1.4
- [7] 神谷泉. 測位技術の調査と IC タグ, UWB の測位への応用. 国土地理院時報, 2005. 2.1.5, 2.1.6
- [8] Z. Sahinoglu, S. Gezici, and I. Güvenc. *Ultra-wideband positioning systems*. Cambridge University Press, 2008. 2.1.5
- [9] H. Kong, Y. Kwon, and T. Sung. Comparisons of TDOA triangulation solutions for indoor positioning. In *international symposium on GNSS, Sydney, Australia*,

- pp. 6–8, 2004. 2.1.5
- [10] F. Armenio, H. Barthel, L. Burnstein, P. Dietrich, J. Duker, J. Garrett, B. Hogan, O. Ryaboy, S. Sarma, J. Schmidt, et al. The epcglobal architecture framework. 2.1.6
- [11] A. Ward, A. Jones, and A. Hopper. A new location technique for the active office. *Personal Communications, IEEE*, Vol. 4, No. 5, pp. 42–47, 2002. 2.1.7
- [12] 渋谷奈保, 高橋友和, 井手一郎, 村瀬洋, 小島祥子, 高橋新. 車載レーザスキャナによる距離データマップの構築と高精度自車位置推定. 電子情報通信学会論文誌, 2009. 2.1.8
- [13] H. Cho, M. Kang, J. Park, B. Park, and H. Kim. Performance Analysis of Location estimation algorithm in ZigBee networks using received signal strength. *Advanced Information Networking and Applications Workshops, International Conference on*, 2007. 2.2.2
- [14] E.E.L. Lau and W.Y. Chung. Accuracy Refinement Algorithm for Mobile Target Location Tracking by Radio Signal Strength Indication Approach. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. 91, No. 7, p. 1659, 2008. 2.2.2
- [15] Y. Takizawa, P. Davis, M. Kawai, H. Iwai, A. Yamaguchi, and S. Obana. Self-organizing location estimation method using received signal strength. *IEICE Transactions on Communications*, Vol. 89, No. 10, p. 2687, 2006. 2.2.2
- [16] 仰木裕嗣. Location Engine を用いたヒトの移動検出に関する研究, 2008. 3.1.1
- [17] 米村茂, 三次仁, 羽田久一, 中村修, 村井純. 移動する ZigBee センサーノードの位置推定アルゴリズム. 電子情報通信学会ソサイエティ大会講演論文集, p. 402, 2009. 3.1.2
- [18] 小澤みゆき, 三次仁, 羽田久一, 白石雅彦, 中村修, 村井純. ZigBee Locaiton Profile における移動センサノードの省電力化プロトコル. 電子情報通信学会ソサイエティ大会講演論文集, p. 386, 2009. 3.1.3