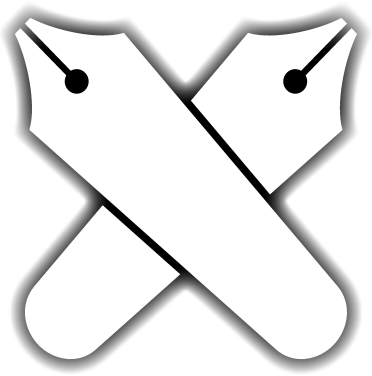
修士論文 2010 年度 (平成 22 年度)

マルチセンサを用いた
リアルタイムサッカー選手分析システムの開発



慶應義塾大学大学院政策・メディア研究科

渡邉　俊

修士論文 2010 年度 (平成 22 年度)

マルチセンサを用いた
リアルタイムサッカー選手分析システムの開発

本研究は，どんな環境でも低コストで簡易的にサッカー選手の分析を可能にするシステム開発を目的とした．その目的を達成するために，GPS，地磁気センサ，ジャイロスコープ，Zigbee を用いて，リアルタイムにサッカー選手分析ができるシステム開発に取り組んだ．本研究では特に，ボールを保持していないオフザボールの選手が『いつ』,『どこで』,『どの向きで』,『なにを』していたか分析することを目標とした．GPS を用いた理由は，サッカーの試合中に選手がグラウンド場の『どこで』プレーしていたのかを分析するためである．また，この GPS を用いることで，選手の移動速度も推定した．地磁気センサを用いた理由は，選手が『どの向きで』プレーしているかを分析するためである．この地磁気センサを用いることで 8 方位の推定を行なった．ジャイロスコープを用いる理由は，選手が『なにを』しているのか分析するためである．オフザボールの運動は，大きく 3 つにわけることができる．それは，歩行，走行，ステップワークである．そこで，ジャイロスコープを大腿部に装着することで，大腿部の内転外転角速度と屈曲伸展角速度を計測し，その計測データを基に歩行，走行，ステップワークの自動運動識別に取り組んだ．この自動運動識別方法には機械学習であるニューラルネットワーク用いた．Zigbee を用いた理由は 2 つある．1 つ目が，各センサから獲得されるデータをリアルタイムに分析するためである．2 つ目が，選手全員の運動計測を時間同期するためである．サッカーは 11 人で行う球技であるため，センサを用いてサッカー選手の『いつ』を分析するためには選手全員の運動計測が時間同期される必要がる．この Zigbee を用いる事で 11 人の同時計測を可能にしたのと同時，1 秒間隔で各選手のプレーをリアルタイムに分析が出来るようにした．

キーワード：
サッカー，GPS，地磁気センサ，ジャイロスコープ，Zigbee

慶應義塾大学大学院政策・メディア研究科
渡邉　俊

Abstract of Master's Thesis Academic Year 2010

Development of a multiple sensor system for realtime analysis of soccer players

The purpose of this research is to develop an efficient, low-cost and robust system which enables performance analysis of soccer players.In the context of this research, a real time multi-component system was developed to achieve such purpose. The system consists of the following components: a GPS, a geomagnetic sensor, a gyroscope and a Zigbee. The motivation behind this research was to analyze performance of off-the-ball players (i.e, players who do not possess the ball). Information of When and Where the location of the players was on the field as well as Which direction they was facing and What they were doing could assist in improving performance analysis of off-the-ball players.The GPS is used to determines the location of the players on the field which in turns gives an estimation of the players speed on the field.The geomagnetic sensor determines the direction the player is facing. The sensor assumes eight different directions.Off-the-ball players' movement could be classified into three main patterns; walking, running and step work. In this research, neural networks is used as an automated pattern recognition mechanism to detect those three patterns. The gyroscope is adjusted on the femur to measure the angular velocity during abduction,adduction, extension and flexion. The output data of the gyroscope is fed into the neural network to determine the movement pattern of the players.A soccer team is made up of 11 players. Therefore, synchronization of the measured data becomes essential in order to understand when each player was moving on the field. The Zigbee module is utilized for data acquisition from the different components of the system as well as synchronizing the information sent from each player on the field. Zigbee makes it possible to perform simultaneous measurement of the data coming from all 11 players. Furthermore, Zigbee enables ,with an interval of 1 second, to perform real time movement analysis of each players on the field.

Key Word :

Soccer, GPS, Geomagnetic sensor, Gyroscope, Zigbee

Keio University Graduate School of Media and Governance

Shun Watanabe

目 次

図 目 次

表 目 次

第1章　緒言

1.1　はじめに

本章は，4つの節から構成される．節 1.2 研究目的では本研究の目的と方法，そして本研究が目指す目標を示す．節節 1.3 指導者の課題では，ジュニアチームの指導者である筆者が強く現場で感じる問題を示す．この問題を解決しようとすることが本研究の動機である．節 1.4 背景では，本研究にいたる 4 つの背景を示す．それは，現代サッカーの流れ，オフザボール[1]の重要性，計測方法，分析方法である．計測方法ではビデオカメラによる計測法およびセンサを用いた計測方法の研究背景について示す．分析方法ではボール中心のゲーム分析，選手中心のゲーム分析に関する研究背景を示す．節 1.5 本論文の筋道では，本章以降の論文構成についてを示す．

1.2　研究目的

本研究の目的は，どんな環境でも低コストで簡易的にサッカー選手の運動分析を可能にするシステムを開発することである．この目的を達成するために，GPS[2]，地磁気センサ[3]，ジャイロスコープ[4]，Zigbee[5]を用いて，リアルタイムにサッカー選手分析ができるシステム開発に取り組んだ．本研究では特に，ボールを保持していないオフザボールの選手が『いつ』,『どこで』,『どの向きで』,『なにを』していたか分析することを目標とした．サッカーを定量的に分析するシステム開発は欧州のサッカー先進国[6]で近年行われている [1]．しかし，そのような分析データを獲得できる環境は，プロチームのような設備投資ができ

[1]ボールを保持していない状況のこと
[2]Global Positioning System
[3]電子コンパス
[4]物体の角速度を検出する計測器
[5]無線通信規格の一つ
[6]イタリア，ドイツ，スペイン，フランス，オランダなどのサッカー強豪国

るようなチームでないと実現されていないのが現状である．そこで，本研究はどんな環境でも低コストで簡易的に選手たちを分析するシステム開発に取り組んだ．

1.3　指導者の抱える課題

筆者が小学生や中学生のサッカー指導者として練習試合を終えたあと，ある保護者から『うちの子をもっとがんばらせてください．いつも歩いてさぼってばかりいる』と言われたことがあった．筆者にとっては，その子は熱心にボールを追いかけ回し，いつも歩いてさぼってばかりいるとは感じていなかった．しかし，我が子を常に観察していたその保護者にとっては，その子が何をしていたのかすべてを観ていたように思える．このため，その評価が間違っているとは言えないだろう．サッカーという球技では，指導者が各選手つまり 11 人のプレーすべてを把握することは非常に難しい．しかし，選手を正当に評価する立場にある指導者にとってそれは大きな問題になる．サッカーは長さ 100〜105m，幅 64〜75m のフィールド場で 前後半 45 分ハーフ合計 90 分間 の試合が行われる．その中で，11 人対 11 人の選手たちが走る，歩く，ステップワーク，ドリブル，パス, シュートといったさまざまな運動を連続的に続けている．サッカーは球技であるため，監督，コーチを含め，観る側はボールに注目することは当然のことである．そのため，各選手の評価が，ボールを保持していた時の印象を非常に受けやすい．しかし，その印象だけで選手を正当に評価できるとは考えられない．なぜなら，選手のパフォーマンスの 90 %以上を無視しているからである．サッカーは２つの時間に分けることができる．それはオンザボールの時間と， オフザボールの時間である．オンザボールとは，ボールを選手が保持している状態のことであり，ドリブルやパス，シュートといった運動のことである．逆にオフザボールとはボールを保持していないときの運動であり，歩き，走り，ステップワークなどがそれにあたる．ただし，90 分間の試合中の中で，サッカー選手一人はどれくらいオンザボールの時間があるのかといえば，実際には２分〜３分である [1]．つまり，ボール中心にサッカーの試合を観察した場合，選手一人あたりの観察時間は数分にしか満たないことになる．それでは，指導者がオフザボールの動きを選手たちに指導することや評価することの妥当性を疑われても仕方ないだろう．ここに指導者へ課題が突き

つけられる．つまり，オフザボールの動きを指導者はどのように観察するかという課題である．サッカーの指導者である筆者がこの課題を解決しようとすることが本研究の動機となった．

1.4　背景

1.4.1　1970 年から現在までの世界サッカーの流れ

2010 年，日本代表チームはワールドカップでベスト 16 という成績をおさめた．この結果は，日本のサッカーレベルが世界の強豪国と競える程のレベルになったことを示した．しかし，日本のサッカーの歴史は世界のサッカー先進国に比べればまだまだ浅いものである．日本がワールドカップに初出場を成し遂げたのは 1998 年のフランス大会である．しかし，第 1 回大会は 1930 年に開催されたウルグアイ大会までさかのぼる．つまり，日本はワールドカップが開催されてから 70 年近く世界のサッカーを肌で感じることができなかった．世界のサッカーの流れは，1930 年のワールドカップによって明確に方向づけられた．しかし，1970 年まで日本のサッカー界の中ではワールドカップはそれほど身近な大会ではなかった．日本サッカーの歴史として，1968 年のメキシコオリンピックで銅メダルを獲得しているが，プロリーグのなかった当時の目標は，アマチュアのチャンピオンになることだった．もちろん，オリンピックの中間年に行われるワールドカップのアジア予選は過去もチャレンジしたが，予選で連続して破れてもドーハの悲劇[7]のようなショックをサッカー界自体も感じてはいなかった．しかし，1970 年の第 9 回メキシコワールドカップから本格的なワールドカップの映像が日本に入り始め，サッカーを志す子供達までもが胸をときめかせるようになってきた．では，1970 年代当時のサッカーはどのようなものだったのであろうか．1970 年代と現在サッカーで最も違う点は，ディフェンスラインとフォワード間の距離である．現在のサッカーでは，ディフェンスラインとフォワードラインを 30 メートル前後に狭めて，この狭いエリアの中でフィールドプレーヤー 20 人が攻防を繰り返している．30 メートルの中でサッカーをやれば，味方同士の距離が近いため，守備におけるカバーリ

[7]カタールのドーハで行われた 1994 年アメリカワールドカップ・アジア地区最終予選の日本代表最終戦において，試合終了間際のロスタイムにイラク代表の同点ゴールが入り，日本の予選敗退が決まった事

ング，攻撃におけるサポートを行いやすい．プレッシャーが厳しくなった分，ボールをもつ選手は以前よりもコントロール[8]のスピードと正確さ，さらに判断の速さが求められるようになった．しかし，1970 年ワールドカップ当時のサッカーは，ディフェンスラインからフォワードラインまでの距離が極端に遠いものであった．サッカーのピッチは，ゴールラインからゴールラインまでが標準で約 105 メートルあるが，フォワードが相手のゴールから 20 メートルの所まで攻め込んでいる場合でも，ディフェンスがまだ自陣の真ん中に残っていることがよく観られた．この頃は，プレーするためのスペースと時間が十分確保されていた．つまり，1970 年代のサッカーは 1 つのボールにたいして多くの選手が関わりをもつ現代サッカーとは異なり，どの場所でも敵味方の 1 対 1 の状況がはっきり把握できた．言い換えれば，現代サッカーは，ボールを保持しているオンザボールの選手から，オフザボールの選手がどのような動きをするかが重要視されるようになった．

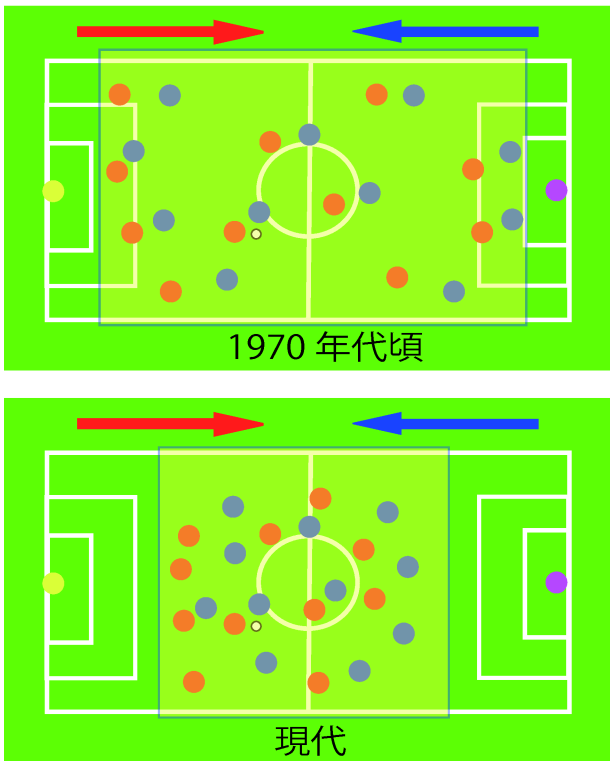


図 1.1: 現代と 1970 年代サッカーの違い

[8]ボールを止める，蹴るの技術

1.4.2　オフザボールの重要性

1970 年代のサッカーから現代サッカーの変化により，ボールを保持しているオンザボールの選手から，オフザボールの選手がどのような動きをするかが重要視されるサッカーに変化した．そのオフザボールの動きは『いつ』,『どこで』,『どの向きで』,『なにを』という要素に分けることができる.『いつ』というのは，ボールの位置や味方選手および敵選手の状況のある時刻を示す．この『いつ』を分析することは選手のある局面でのプレーの要因を探ることになる.『どこで』というのは，選手がグランド上のどこに位置をとっていたのかを示す．サッカーの指導者が，試合中選手に対して一番指示することはポジショニングつまり『どこで』プレーするべきかという事である．サッカーは集団スポーツであることから，選手一人の位置が変われば，残り 21 人の最適な位置取りは変わってくる．つまり,『どこで』を分析することは，得点，失点およびオンザボール選手のプレー成功，失敗の原因追求につながる.『どの向きで』というのは，選手がどの方向に体を向けていたのかを示す．サッカーの試合中，選手の体の向きはその選手が次にどのようなプレーができるかを示す重要な情報となる．たとえば，図 1.2 に示すケース 1 とケース 2 の状況を考えてみる．

ケース 1 とケース 2 の違いは選手 A からパスを受けようとする選手 B の体の向きである．ケース 1 では選手 B は体の向きがゴールの方向に向いているため，選手 A からのパスをゴールに近い位置で受けることができる．よりゴールに近い位置でボールを受けることができれば，それだけゴールする可能性も高くなる．しかし，ケース 2 では選手 B はボールの方向に体が向いているため，ケース 1 よりゴールから離れた位置でパスを受けることになる．このように，オフザボールの選手の体の向きが違うだけで，次のプレーの状況が大きく異なる．このため,『どの向きで』の要素は非常に重要であると考えられる.『なにを』というのは，選手がどんな運動をしていたのかということを示す．その運動は大きく 3 つに分けることができる．それは，歩行，走行，ステップワーク[9]である．この『なにを』を分析することは，選手一人のパフォーマンス[10]評価につながる．以上のように，サッカー選手のオフザボールの動きを分析するためには,『いつ』,『どこで』,『どの向きで』,『なにを』の分析をする必要がある．そこで，本研究では,『いつ』の分析のために Zigbee を,『どこで』の分析

[9]サッカーにおける歩行，走行以外の移動動作
[10]どのくらい試合中働いたか

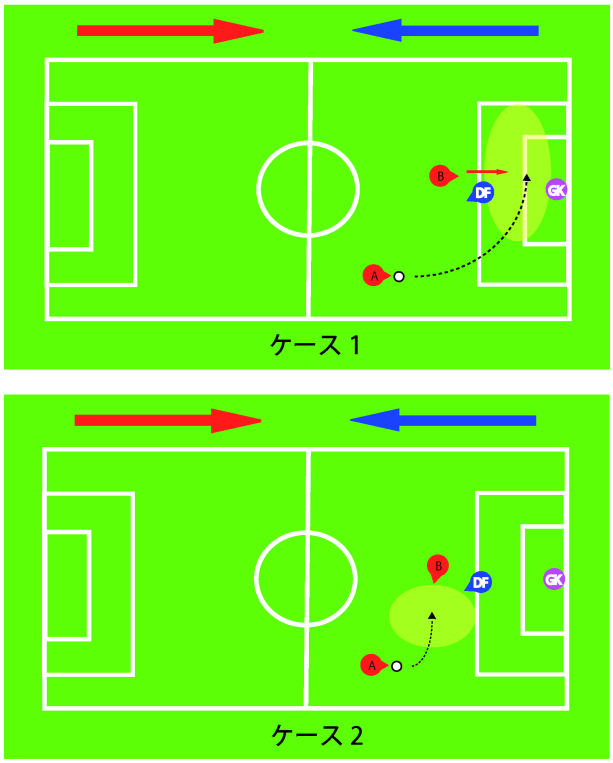


図 1.2: オフザボールにおける体の向きの違い

のために GPS を,『どの向きで』の分析のために地磁気センサを,『なにを』の分析のためにジャイロスコープを用いることにした.

1.4.3 計測方法

ビデオカメラ

現在のサッカーのゲーム分析や個々の選手パフォーマンスを分析するための計測方法として主流は，ビデオカメラを用いることである [7]．しかし，ビデオカメラ撮影は，撮影する環境に依存する．ビデオカメラを 1 台だけ用いて，約 105 × 70m のグラウンド全体を撮影しようとするのであれば，非常に高い位置に設置できる場所がなければならない．それが出来なければ，ビデオカメラの向きを調整し，ボール中心にゲームを撮影することになる．そうなれば，カメラの画角内に映った場所や選手の記録しか残らず，試合中選手すべての動きを記録できない．そこで近年，固定カメラを複数用いてそれらを同期し，DLT 法による 3 次元化および画像分析をもちいて選手を自動認識できるシステムが開発されている [10]．今年開催したワールドカップ南アフリカ大会では

TRACAB 社が開発したビデオカメラシステムにより出場選手全員のデータが記録され FIFA.com で公開されている [12]．しかし，このようなビデオカメラシステムを用いるには莫大なコストがかかることや，スタジアムのような限られた環境でしか計測できない．つまり，プロサッカーチームやナショナルチームのような限られたチームだけがスタジアムを保有し，数千万から数億円のシステムを導入できる．アマチュアやジュニアのチームのような設備投資が困難なチームにとっては，低コストで計測環境に依存しない方法が必要である．

センサ

近年小型センサが非常に低コストで手に入るようになった. また, そのような小型センサを身体部に装着し, 身体運動の分析をする研究も近年盛んに行われている [13]．スポーツに関する研究でこのような小型センサを身体に装着する大きなメリットは 2 つある．1 つ目は加速度などからの運動学情報が直接検出できるため，画像分析よりも精度の高い情報を獲得できることである．2 つ目が，計測環境に依存しないことである．また，このような小型センサモジュールがマイクロコンピュータと接続され無線で制御通信できるようになれば，複数のセンサモジュールの時間を同期することができる．例えば，球技などの集団競技で各選手がこのようなセンサを装着した場合，選手全員の運動を同時計測することが可能になる．このような小型センサは年々その性能，コスト，大きさが劇的に改善されてきている．そこで，本研究ではこのような小型センサを用いてサッカー分析システムを構築する．

1.4.4 分析方法

ボール中心のゲーム分析

22 人もの選手がいて，それも敵味方入り乱れて動きを止めないとなれば何か基準を決めないと分析は成り立たない．フィールドプレーヤーは 20 人いるが，ボールは 1 つある．そこで，このボールを基準にしてプレー分析をする試みが過去盛んに行われ，現在でも試合分析の中心になっている [15]．このような分析では，ボールの動き，たとえばパスの分析ではパソコンを使ったリアルタイム分析システムが実用化されている．ボール保持時間，パス連続頻度，選

手間のパス頻度を解析するために，データ入力は通常 2 名で行う．一人は試合を観察しながら，ボールを保持する選手の背番号とプレー内容を口述する．もう一人は，選手とプレーに対応したキーボタンから入力を行う．このような方法で，リアルタイムに情報が入力され，ある区切られた期間，たとえばハーフタイムにプリンタからの出力が行われ，それが現場にフィードバックされる．しかし，そのようなボール中心の分析だけでは結果だけしか記録されていないのと同じである．つまり，何点ゴールが決まったかを記録しているのと同じ事である．確かに，パスの成功率や試合中のボール支配率[11]を推定する事は，その試合がどのような内容だったのかを指す指標となる．しかし，指導者や選手にとって重要なのはパスが成功したか失敗したという結果ではなく，なぜ成功したのか，なぜ失敗したのかという原因の追求である．この原因を解明し解決方法を示すことこそが指導者の任務である．パスは出し手の技術だけではなく受け手の位置や体の向き，また周りの選手のサポートが適切に行われた時に成功する．このため，ボール中心だけの分析だけではパスやシュートの成功，失敗の原因を示すことはできない．そこで，それぞれの選手がボールのない時にどのような反応をしているのか，ボールの位置によってどのように動きを変えているか，いつ動きだしているか，さらにボールを持った時のプレーの成功をもたらした鍵になるオフザボールの動きは何だったか，ボールをもっていない時の動きを分析する必要がある．

選手の視点からのゲーム分析

選手が 20 人いても，1 人の選手に焦点を当てれば，分析が容易である．この観点からの研究も行われている [17]．しかし，選手一人がいつ，どこで，なにをしたのかを分析しただけでは，その選手がなぜその判断をしたのかを追求することはできない．重要なのは，敵の位置，ボールの位置，味方の位置がどのような状態のとき，その選手がどのような判断をしたかである．1990 年代後半のサッカーでは 1 対 1 の局面が多数現れていたため，各選手の判断は個人の状態に委ねられていた [2]．つまり，選手一人だけを分析するだけでも十分だったといえる．しかし，現代サッカーでは，1 対 1 の局面がうまれることは稀である．必ず複数人の動きが関連して一人のプレーヤーの動き，位置，プレーの

[11] どちらのチームがどのくらいボールを保持していたかを示す指標

選択が決まってくる．このため，ボール中心の分析と同時に選手全員の動きを分析する必要がある．

1.5　本論文の道筋

本章以降において本論文は以下の内容によって構成される．

2章　開発方法： 開発方法ではまず本研究で目指す開発システム概要と，試作したハードウェアとソフトウェアについて述べる．そして，実際に開発するプロトタイプについて述べる．

3章　計測： 製作したプロトタイプを用いて，選手が,「いつ」,「どこで」,「どのような」の動きをしているのかを計測する方法を示す.「いつ」,「どこで」の計測ではGPSによる位置計測を行い分析する.「いつ」,「どのように」の計測は地磁気センサとジャイロスコープセンサを用いる．地磁気センサにより体の向きを検出する．ジャイロスコープによる計測は，選手が歩いているのか，走っているのか，ステップワークをしているのかを分析するために用いる．

4章　運動識別： 大腿部に装着したジャイロスコープから得られる角速度データを基に，選手が歩いているのか，走っているのか，ステップワークをしているのかの分析をコンピューターが自動的に運動識別を行う方法を示す．この運動識別にはニューラルネットワークを用いた．ニューラルネットを用いる目的は，運動識別をコンピューターが自動的に行うための最適なパラメーターを獲得するためである．

5章　考察： 開発したプロトタイプを用いたフィールド実験家についての考察を示すと共に，この開発したシステムが実際のサッカーの現場でどのように用いることができるのかについての考察を行う．

6章　結言： 本研究の総括と今後の展望を述べる．

1.6　まとめ

サッカーでは一人の選手がボールに触れる時間は多い選手で2分から3分である．つまり，一人一人の選手をみればオンザボールの局面よりオフザボールの局面の方が圧倒的に多いということである．それぞれの選手がボールがない時にどのような反応をしているのか，ボールの位置によってどのように動きを変えているか，いつ動きだしているか，さらにボールを持った時のプレーの成功をもたらした鍵になるオフザボールの動きは何であったのかなど，ボールをもっていない時の動きの分析ができるシステム開発することは，サッカーの質的向上におおいに役立つ．近年そのようなシステム開発はビデオカメラを複数台設置し，画像分析から行われることが主流である．しかし，そのようなシステムは高性能なビデオカメラ，画像分析処理を用いるため数千万，数億円規模の費用がかかる．また，スタジアムのような適切なビデオ設置位置がない場所では計測することができない．そのようなシステムを用いることが出来るのはナショナルチームやプロチームのような設備投資ができるチームでしか使用することができないのが現実問題である．そこで，低コストで，計測環境に依存しない小型センサを用いたシステム開発に取り組んだ．

第2章　開発方法

2.1　はじめに

本章ではまず本研究で目指す開発システム概要を示す．そして，開発するためのハードウェアとなるセンサモジュール，制御コンピュータ，ソフトウェアについて述べる．用いるセンサモジュールは，GPS，ジャイロスコープ，地磁気センサである．そして，開発した計測器のプロトタイプ と計測 PC について述べる．

2.2　開発システムの概要

近年マイクロコンピュータや小型センサが非常に低コストで手に入るようになった．このようなマイクロコンピュータや小型センサ，Zigbee，ビデオカメラを用いてシステム開発を行う．サッカー選手がオフザボール時に『いつ』,『どこで』,『どの向きで』,『なに』をしているのかを分析するために本研究では 3 つのセンサを用いる．それら 3 つのセンサは GPS，地磁気センサ，ジャイロスコープである．GPS を選手に装着することで，ある時間の選手の位置を推定し,『どこで』の分析を行う．そして，選手の位置を推定することで，同時刻の速度，移動距離，移動軌跡も推定する．地磁気センサは方位，つまり『どの向きで』を分析するために用いる．地磁気センサを選手に装着することでその選手がどの方角に向いていたのかを推定する．ジャイロスコープは，選手がオフザボール時に『なにを』しているのかを識別するために用いる．その識別方法に機械学習であるニューラルネットワークを用いる．ニューラルネットワークを用いた運動識別については第 4 章で詳しく述べる．これら 3 つのセンサを制御するためのコンピュータとして mbed を用いる．複数の選手を同時に計測するため，本研究では Zigbee を用いる．これによりオフザボールの『いつ』の分析が可能になる．さらに，Zigbee を用いることで，計測データは無線でリア

ルタイムに計測 PC に送られ，リアルタイムな分析を可能にする．ボールがどのような状態なのか，つまりどの選手がボールを保持しているのかを計測するめにはビデオカメラを用いる．どのように選手がボールを保持しているのかという詳細な情報はサッカー分析では非常に重要になる．たとえば，保持している選手が右足でボールを保持しているのか，左足で保持しているのかの違いだけでも，オフザボールの動きやその位置どりが変わってくる．このようなボール保持状態を詳細に識別することはセンサだけでは非常に困難であった．そのため，オンザボールの状態は映像を用いて記録することにする．この映像情報は，選手に装着した各センサ情報と計測 PC で同期される．計測 PC がインターネットに繋がっていれば，計測されたデータはデータベースサーバーに送信される．そして，データベースに保存されたデータは web アプリケーションとして可視化される．web アプリケーションとして提供することで, 計測データを携帯電話などのモバイル端末からも見る事が可能になる. つまり, 試合後および練習後に指導者が選手のプレーを即座に分析できる．システム概要を図 2.1 に示す.

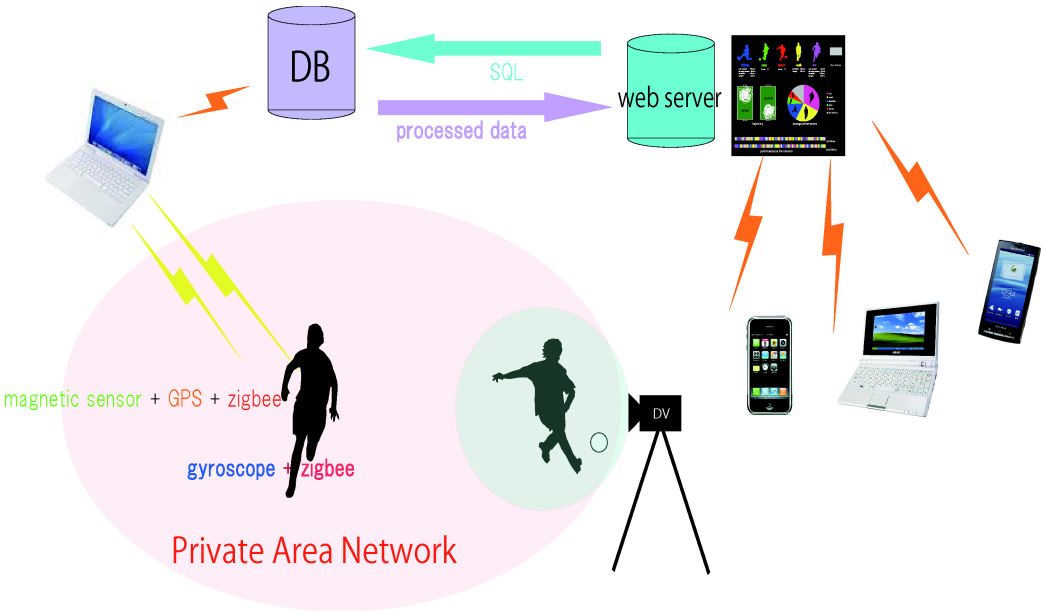


図 2.1: システム概要図

2.3　制御用マイクロコントローラ

各センサを制御する目的で, 本研究では mbed を用いる．mbed は NXP LPC1768 チップを搭載した DIP40pin の高速プロトタイピング向けマイクロコントローラボードである．図 2.2 に mbed の概観図を示す．

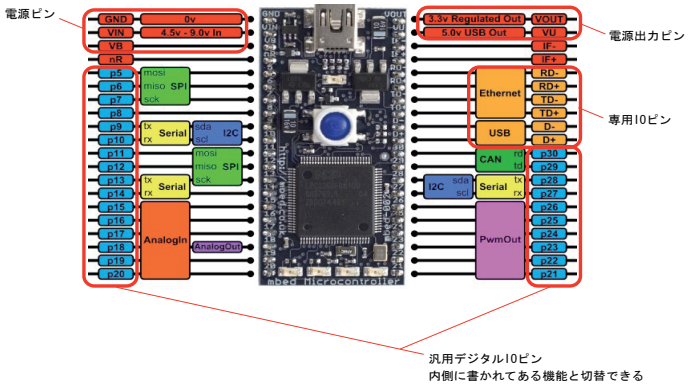


図 2.2: mbed

(出典：http://mbed.org/media/uploads/nxpfan/mbed_fest_v1_print.pdf)

2011 年 1 月 11 日

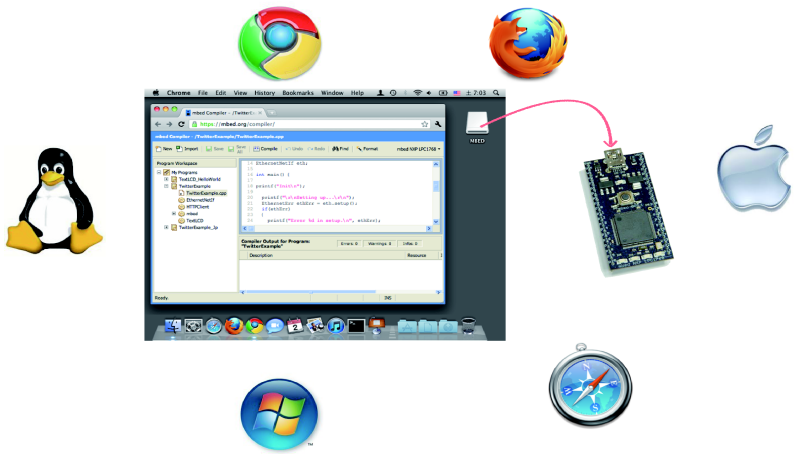


図 2.3: mbed の開発環境

(出典：http://mbed.org/media/uploads/nxpfan/mbed_fest_v1_print.pdf)

2011 年 1 月 11 日

図 2.3 が示すように開発環境は，C/C++を用いる．mbed の特徴は，開発環境がすべて web 上に存在することである．つまり，PC で開発環境を整えるためソフトウェアのインストールは不要であり，インターネットに繋がった PC さえ

あればいつでも開発ができる．mbed と同じようマイクロコントローラボードとして，ATmega328P を搭載したマイクロコントローラボードの Arduino[1]がある．本研究では開発当初，この Arduino を用いて開発を行っていた．しかし，Arduino と比較し mbed は，接続インターフェースが充実し，マイコンボードの性能も格段に高いため本研究では mbed を採用した．Arduino と mbed の特徴を表 2.1 に示す．mbed のハードウェアについては図 2.5 に示す．mbed のチップである LPC1768 のブロック図を図 2.4 に示す．mbed のソフトウェアについては図 2.6 に示す.

表 2.1: Arduino と mbed の比較

	Arduino	mbed
Microcontroller	ATmega328P	ARM Cortex-M3
Operating Voltage	1.8V〜5.5V	4.5V〜9.0V
Flash Memory	32KB	512KB
RAM	1KB	64KB
Clock Speed	8MHz	96MHz

[1]単純な入出力を備えた基板と Processing/Wiring 言語を実装した開発環境から構成されるマイクロコントローラーボード

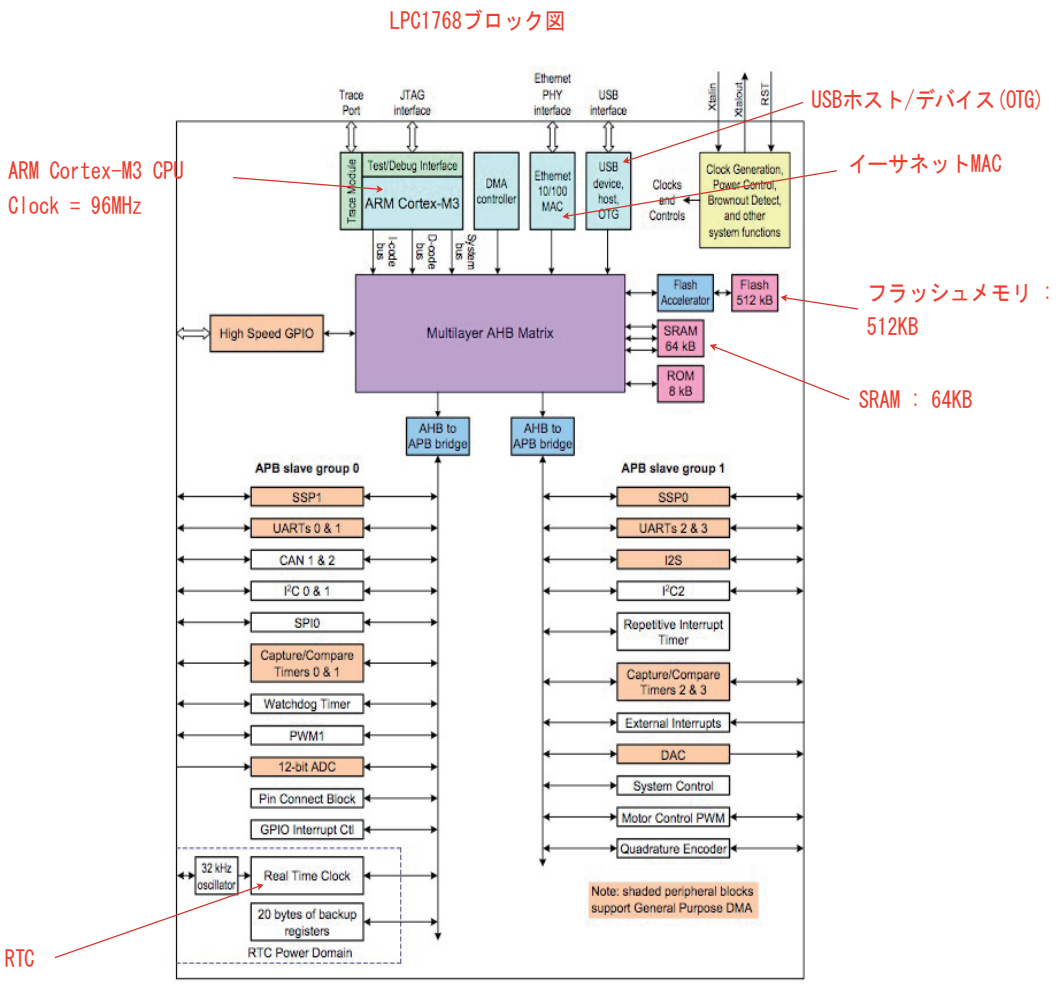


図 2.4: LPC1768 のブロック図

(出典： http://mbed.org/media/uploads/nxpfan/mbed_fest_v1_print.pdf)

2011 年 1 月 11 日

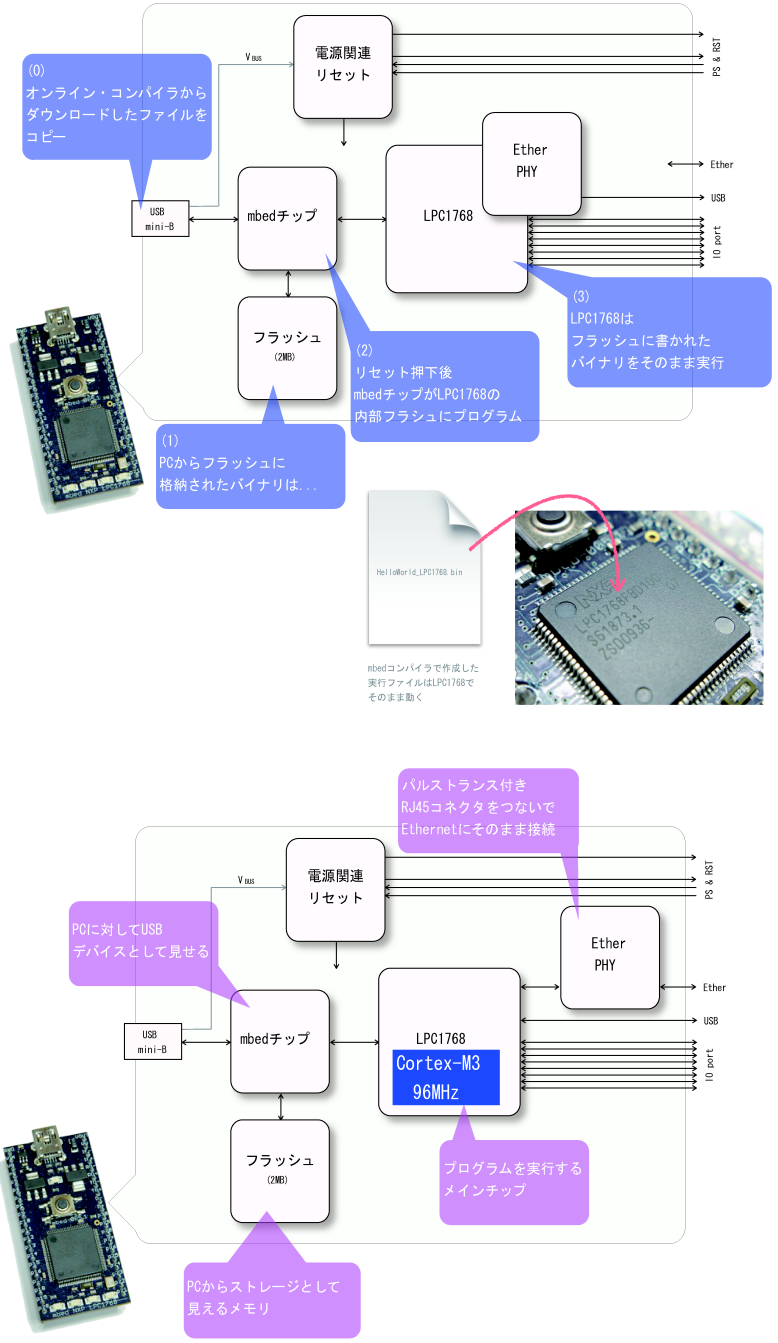
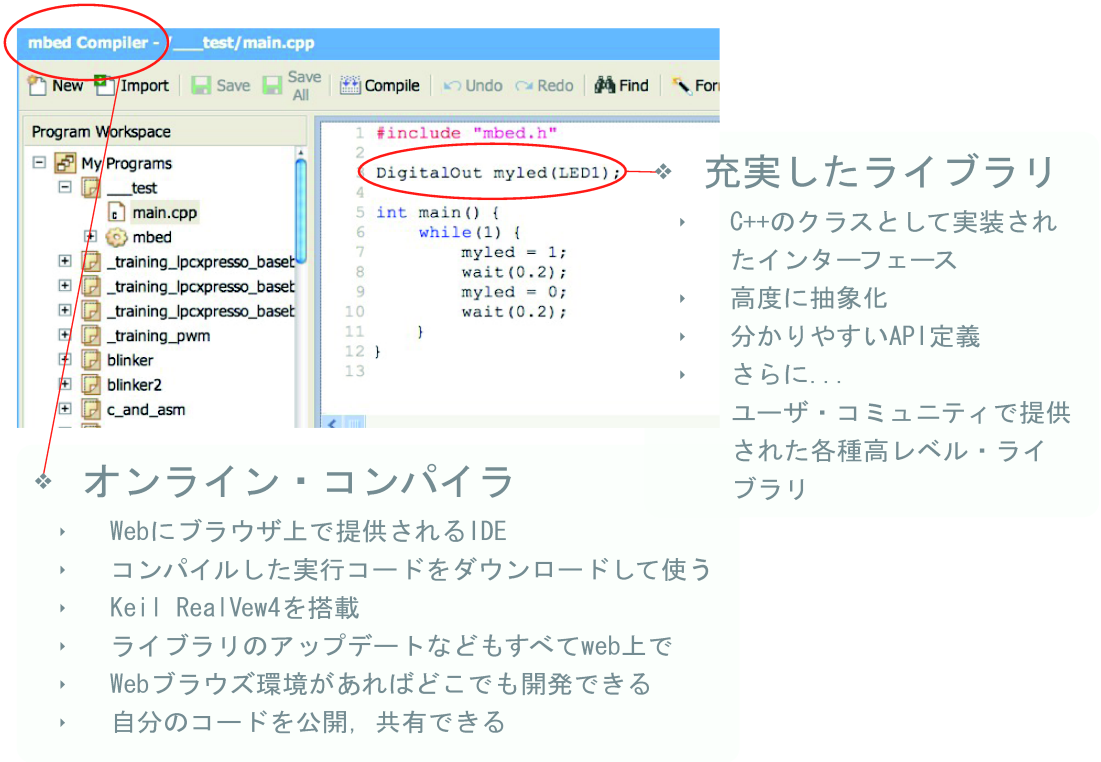


図 2.5: mbed ハードウェア

(出典： http://mbed.org/media/uploads/nxpfan/mbed_fest_v1_print.pdf)

2011 年 1 月 11 日



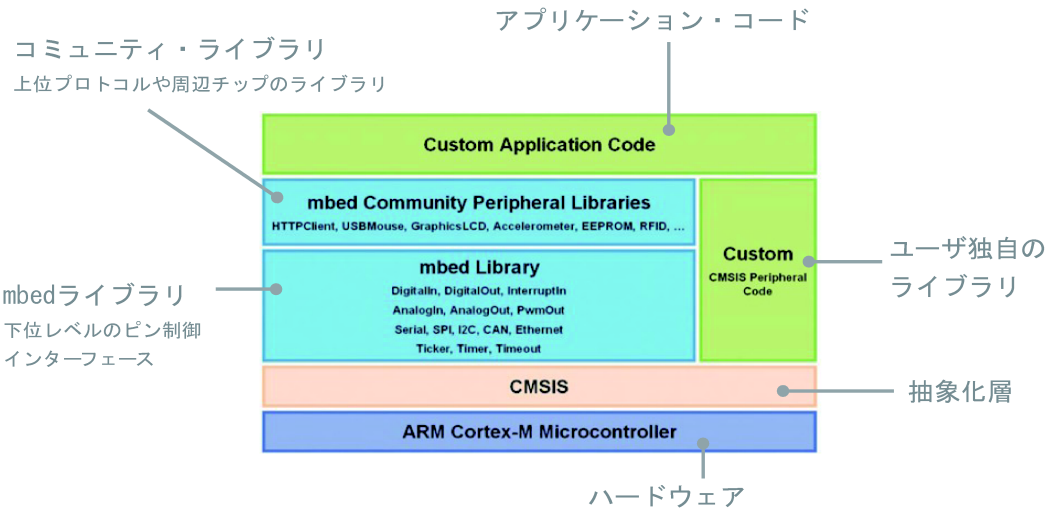


図 2.6: mbed ソフトウェア

(出典：http://mbed.org/media/uploads/nxpfan/mbed_fest_v1_print.pdf)

2011 年 1 月 11 日

2.4 Zigbee

本研究においてZigbeeを使う理由は２つある．一つ目はフィールド上の選手たちに装着したセンサから，データを無線でリアルタイムに獲得することである．二つ目は全選手の計測データを時刻同期することである．Zigbeeを使うために，Digi International Incが開発したZigbeeモジュールXBee Series 2を用いた．Xbee Series 2は，Zigbee規格に基づいて通信を行うモジュールである．図2.7にZigbeeの概観図を示す．

図 2.7: XBee Series 2モジュール

Zigbeeは短距離無線通信規格の一つであり，低速で転送距離が短いが, 安価で消費電力が少ないのが特徴である．データ転送速度は20Kbps〜250Kbpsで, 使用する無線周波数帯によって異なり, 2.4GHzでは250Kbps, 902〜928MHzでは40Kbps, 868〜870MHzでは20Kbpsとなる．900MHz帯を用いたものは主に米国向けで, 800MHz帯を用いたものは主に欧州向けの仕様である．Zigbee端末には中継機能（マルチホップ）があり, 中継を繰り返す事でZigbee素子同士が通信を行える限り情報を伝える事が出来る．1つのPAN[2]内には, 最大で65535個 (アドレスで0x0000〜0xFFFF) のZigbee端末を接続することが出来る．Zigbee端末は図2.8に示すように以下の３種類に分類される．それは，Coordinator, Router, End Deviceである．

[2]Personal Area Network

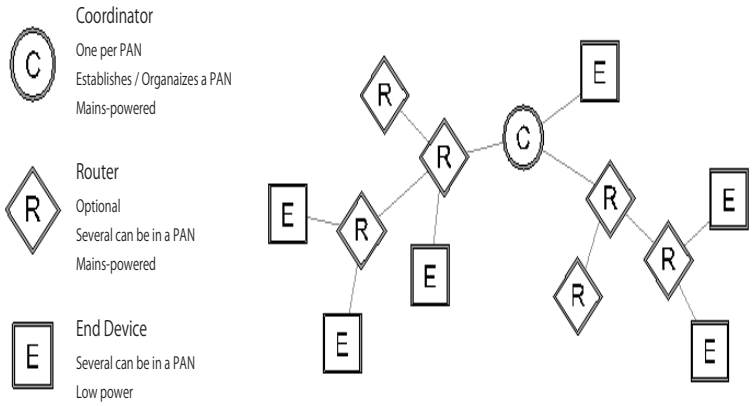


図 2.8: ネットワーク構成図

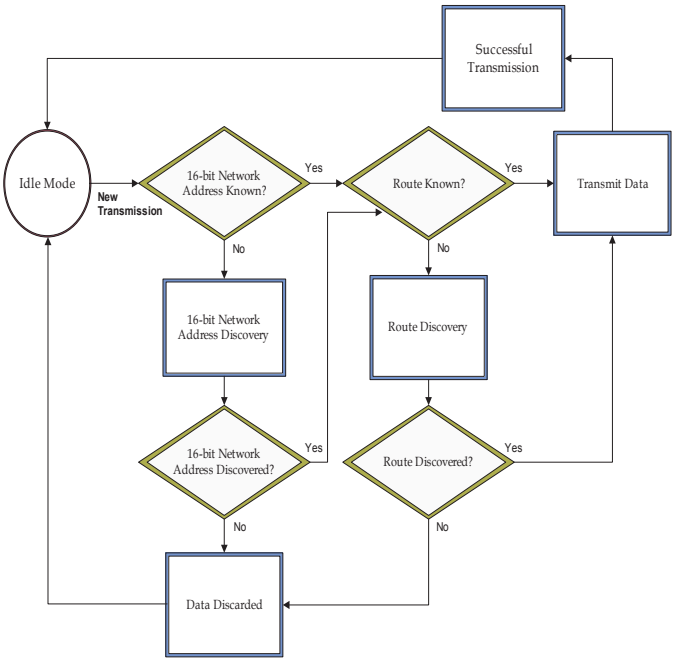
(出典：ftp://ftp1.digi.com/support/documentation/90000976_C.pdf)

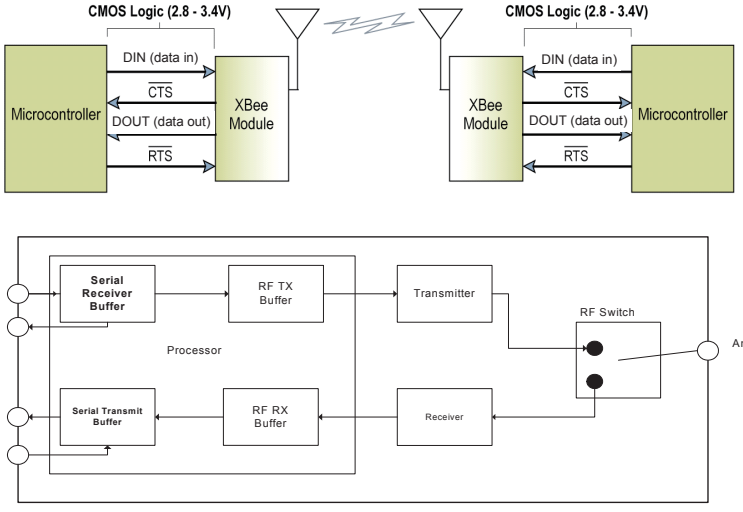
2011 年 1 月 11 日

Coordinator はネットワーク内に 1 台存在し, ネットワークの制御を行う端末である．Router はデータ中継機能を含む端末である．End Device はデータ中継機能を持たない端末である．そのため, 本研究ではこの End Device を選手たちに装着する．Zigbee の特徴は, メッシュ型やツリー型のネットワークを構成し，Router がデータを中継することで, 直接電波の届かない端末間でも通信が可能になる点にある．これにより一部の端末が停止した場合でも, 迂回経路を使って通信を継続できる他, 低消費電力で広範囲で通信を行う事が出来る．そこで，メッシュ型ネットワークをサッカー場に構築する．メッシュ型ネットワークを構築する目的は複数人の選手を同時計測するためである．Xbee Series 2 の設定を表 2.2 に示す．Xbee に設定した UART 通信方法は図 2.9 に示す．

表 2.2: Xbee モジュールの設定

Firmware version	2141
Serial port baud rate	9600bps
Transmit rate	1Hz
Transmit mode	UART





The $\overline{RTS}$ and $\overline{CTS}$ module pins can be used to provide $\overline{RTS}$ and/or $\overline{CTS}$ flow control. $\overline{CTS}$ flow control provides an indication to the host to stop sending serial data to the module. RTS flow control allows the host to signal the module to not send data in the serial transmit buffer out the uart. $\overline{RTS}$ and $\overline{CTS}$ flow control are enabled using the D6 and D7 commands.

図 2.9: UART 通信方法図

(出典：ftp://ftp1.digi.com/support/documentation/90000976_C.pdf)

2011 年 1 月 11 日

2.5 GPS

本研究で GPS を用いる理由は, 各選手の位置と速度を算出することである. GPS のモジュールとして sup500f を用いることにした. 図 2.10sup500f の概要図を示す.

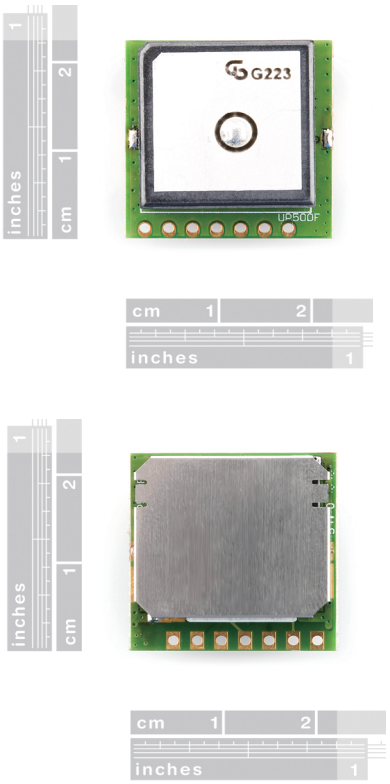


図 2.10: sup500f モジュール

GPS は人工衛星を利用した位置検出システムである. 米軍の軍事技術の一つで, 地球周回軌道に３０基程度配置された人工衛星が発信する電波を利用し, 受信機の緯度, 経度, 高度などを数 cm から数十 m の誤差で割り出すことができる. しかし, 民間用に利用できるデータは, 故意に精度が落とされている. このため誤差は 10m 程度となる. そこで誤差をより小さくするため, 正確な位置の分かっている地上の基準局から電波を発信し, これを利用して位置情報の補正を行なう DGPS[3] という技術が実用化されており, 誤差を数 m に縮めること

[3]Differential GPS

ができる．また, 静止衛星型衛星航法補強システム SBAS[4]を使うことで, 誤差を補正することができる．SBAS は静止衛星からの補正データを利用した高精度の測位システムである．本研究で使用する sup500f にも SBAS を用いることができ誤差を 2m 程度に補正することができるとされている． sup500f の特徴は表 2.3 に示す．その設定については表 2.4 に示す．sup500f で設定したデータ型 NMEA[5](RMC[6]) については図 2.11 に示す．Sup500f のブロック図を図 2.12 に示す．

RMC – Recommended Minimum Specific GNSS Data
Time, date, position, course and speed data provided by a GNSS navigation receiver.

Structure:
$GPRMC,hhmmss.sss,A,dddmm.mmmm,a,dddmm.mmmm,a,x.x,x.x,ddmmyy,,,a*hh<CR><LF>
1 2 3 4 5 6 7 8 9 10 11
Example:
$GPRMC,111636.932,A,2447.0949,N,12100.5223,E,000.0,000.0,030407,,,A*61<CR><LF>

Field	Name	Example	Description
1	UTC time	0111636.932	UTC time in hhmmss.sss format (000000.000 ~ 235959.999)
2	Status	A	Status 'V' = Navigation receiver warning 'A' = Data Valid
3	Latitude	2447.0949	Latitude in dddmm.mmmm format Leading zeros transmitted
4	N/S indicator	N	Latitude hemisphere indicator 'N' = North 'S' = South
5	Longitude	12100.5223	Longitude in dddmm.mmmm format Leading zeros transmitted
6	E/W Indicator	E	Longitude hemisphere indicator 'E' = East 'W' = West
7	Speed over ground	000.0	Speed over ground in knots (000.0 ~ 999.9)
8	Course over ground	000.0	Course over ground in degrees (000.0 ~ 359.9)
9	UTC Date	030407	UTC date of position fix, ddmmyy format
10	Mode indicator	A	Mode indicator 'N' = Data not valid 'A' = Autonomous mode 'D' = Differential mode 'E' = Estimated (dead reckoning) mode 'M' = Manual input mode 'S' = Simulator mode
11	checksum	61	

図 2.11: RMC データ型

(出典:http://www.sparkfun.com/datasheets/GPS/Modules/SUP500F_v3.pdf)

2011 年 1 月 11 日

[4]Satellite-based Augmentation System

[5]National Marine Electronics Association が定めた受信機とナビゲーション機器の通信に使用されるプロトコル

[6]Recommended Minimum Specific GNSS Data

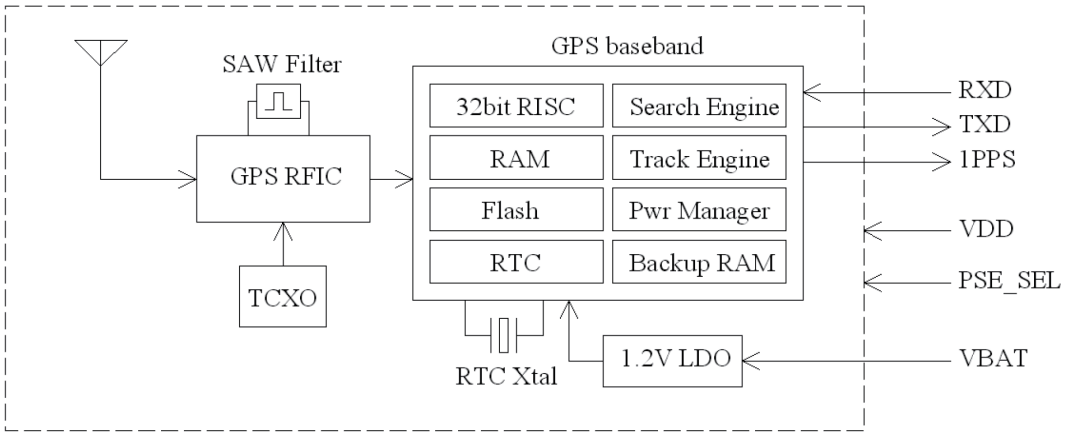


図 2.12: sup500f ブロック図

(出典:http://www.sparkfun.com/datasheets/GPS/Modules/SUP500F_v3.pdf)

2011 年 1 月 11 日

表 2.3: sup500f の特徴

(出典:http://www.sparkfun.com/datasheets/GPS/Modules/SUP500F_v3.pdf)

2011 年 1 月 11 日

Receiver Type	L1 C/A code, 65-channel Venus 6 engine
Accuracy	Position 2.5m CEP Velocity 0.1m/sec Time 300ns
Startup Time	1 second hot start under open sky 29 second cold start under open sky (average)
Reacquisition	1s
Sensitivity	-161dBm tracking
Multi-path Mitigation	Advanced multi-path detection and suppression
Update Rate	Supports 1 / 2 / 4 / 5 / 8 / 10 Hz update rate (1Hz default)
Dynamics	4G (39.2m/sec^2)
Operational Limits	Altitude < 18,000m or velocity < 515m/s
Serial Interface	3V LVTTL level
Protocol	NMEA-0183 V3.01 GPGGA, GPGLL, GPGSA, GPGSV, GPRMC, GPVTG*1 9600 baud, 8, N, 1
Datum	Default WGS-84 User definable
Input Voltage	3.0V ~ 5.5V DC
Input Current	~33mA tracking
Dimension	22mm L x 22mm W
Weight:	9g
Operating Temperature	-40^{o}C ~ +85^{o}C
Storage Temperature	-55 ~ +100^{o}C
Humidity	5% ~ 95%

表 2.4: sup500f の設定

Serial port baud rate	9600
Output message	NMEA(RMC)
Sampling rate	1Hz

2.6　地磁気センサ

本研究で地磁気センサを用いる理由は，各選手の体がどの方向に向いて運動をしているかを検知するためである．地磁気センサは，地球が持つ磁気及び地球上に生じる磁場を感知する．つまり方位を推定するためのセンサである．そこで，三軸デジタル地磁気センサモジュールである HMC5842 を用いる．HMC5843 モジュールを図 2.13 に示す．HMC5843 の特徴は表 2.5 に示し，設定については表 2.6 に示す．HMC5843 のブロック図は図 2.14 に示す．

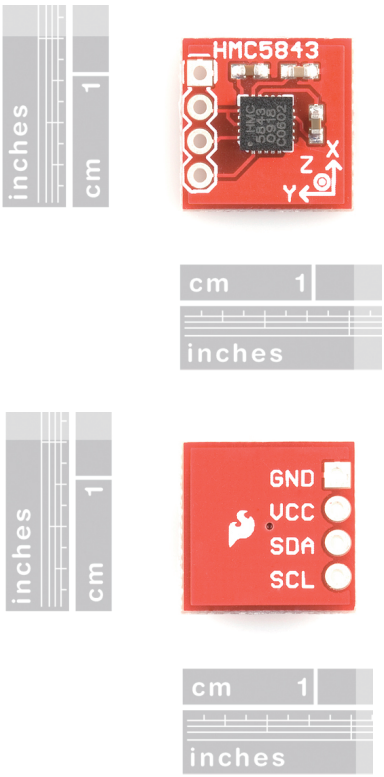


図 2.13: HMC5843 モジュール

表 2.5: HMC5843 の特徴

(出典:http://www.magneticsensors.com/datasheets/HMC5843.pdf)

2011 年 1 月 11 日

SPECIFICATIONS (* Tested at 25°C except stated otherwise.)

Characteristics	Conditions*	Min	Typ	Max	Units
Power Supply					
Supply Voltage	AVDD Referenced to AGND	2.5		3.3	Volts
	DVDD Referenced to DGND	1.6	1.8	2.0	Volts
Current Draw	Sleep Mode (dual supplies)	-	2.5	-	uA
	Idle Mode (dual supplies)	-	240	-	uA
	Measurement Mode	-	0.8	-	mA
	AVDD = 2.5 volts, DVDD = 1.8 volts				
	Sleep Mode (single supply)	-	110	-	uA
	Idle Mode (single supply)	-	340	-	uA
	Measurement Mode	-	0.9	-	mA
	AVDD = 2.5 volts				
Performance					
Field Range	Full scale (FS) – total applied field	-4		+4	gauss
Cross-Axis Sensitivity	Cross field = 0.5 gauss, Happlied = ±3 gauss		±0.2%		%FS/gauss
Disturbing Field	Sensitivity starts to degrade. Use S/R pulse to restore sensitivity.			20	gauss
Max. Exposed Field	No perming effect on zero reading			10000	gauss
Measurement Period	Output Rate = 50Hz (10Hz typ.)		-	10	msec
I^2C Address	7-bit address		0x1E		hex
	8-bit read address		0x3D		hex
	8-bit write address		0x3C		hex
I^2C Rate	Controlled by I^2C Master	-10		+10	%
I^2C bus pull-up	Internal passive resistors		50		kilo-ohms
I^2C Hysteresis	Hysteresis of Schmitt trigger inputs on SCL				
	and SDA - Fall (DVDD=1.8V)		0.603		Volts
	Rise (DVDD=1.8V)		1.108		Volts
Self Test	Positive and Negative Bias Mode		±0.55		gauss
Mag Dynamic Range	3-bit gain control	±0.7	±1.0	±4.0	gauss
Linearity	Full scale input range			0.1	±% FS
Gain Tolerance	All gain/dynamic range settings		±5		%
Bandwidth	-3dB point		10		kHz
Resolution	AVDD=3.0V, GN		7		milli-gauss
Signal-to Noise Ratio		70			dB
Turn-on Time			200		us
General					
ESD Voltage				700	V
Operating Temperature	Ambient	-30		85	°C
Storage Temperature	Ambient, unbiased	-40		125	°C
Weight	Nominal		50		milli-grams

表 2.6: HMC5843 の設定

Measurement range	± 1Ga
Sampling rate	1Hz

INTERNAL SCHEMATIC DIAGRAM
HMC5843

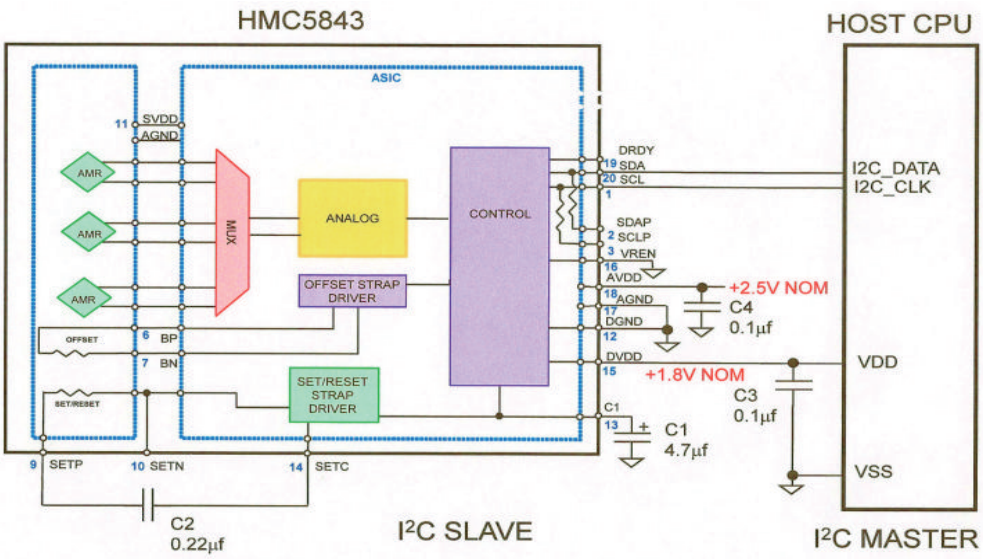


図 2.14: HMC5843 ブロック図

(出典:http://www.magneticsensors.com/datasheets/HMC5843.pdf)

2011 年 1 月 11 日

2.7　ジャイロスコープ

本研究においてジャイロスコープを用いる目的は, 各選手がどのような運動をしているか識別することである. センサモジュールは二軸デジタルジャイロスコープモジュールである LPY5150AL を用いる. 図 2.15 に LPY5150AL の概要図を示す.

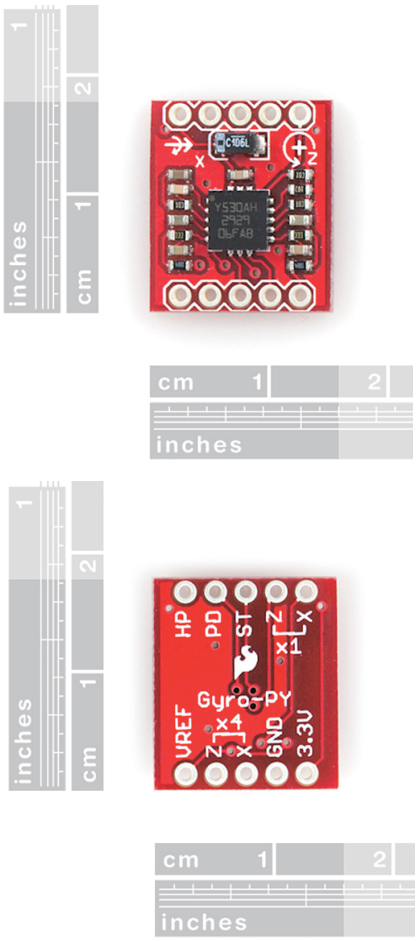


図 2.15: LPY5150AL モジュール

ジャイロスコープは, 物体の回転角速度を検出するセンサである．人間の身体部に装着した場合, 装着したセグメントの回転角速度を検出することができる．そこで, このジャイロスコープを図 2.16 のように右大腿部に装着し，大腿部の内外転，屈曲伸展の回転運動を計測する．計測したデータを基に静止，歩き，走り，ステップワークの自動運動識別を行う．LPY5150AL の特徴は表 2.7 に示し，LPY5150AL の設定については表 2.8 に示す．LPY5150AL のブロック図は図 2.17 に示す．

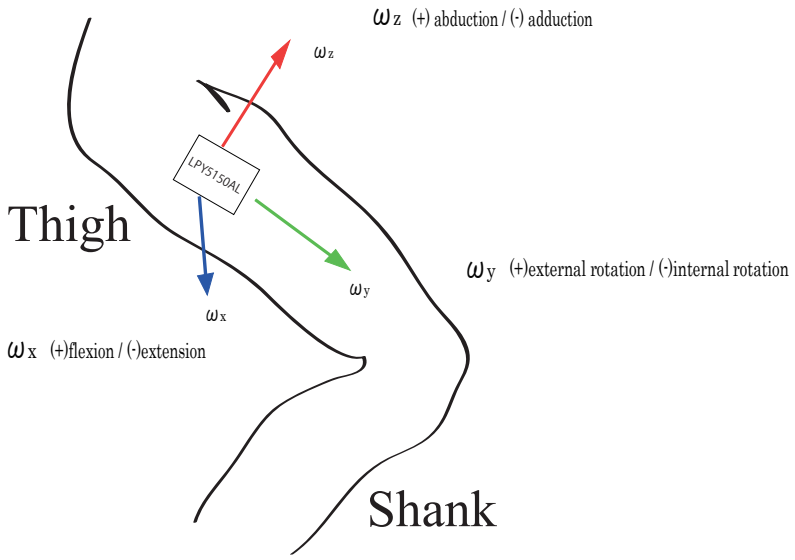


図 2.16: LPY5150AL の装着位置

表 2.7: LPY5150AL の特徴

(出典：http://www.sparkfun.com/datasheets/Sensors/IMU/lpy5150al.pdf)

2011 年 1 月 11 日

Parameter	Test condition	Min.	Typ.(2)	Max.	Unit
Measurement range	4x OUT (amplified)		±1500		°/s
	OUT (not amplified)		±6000		°/s
Sensitivity(3)	4x OUT (amplified)		0.67		mV/ °/s
	OUT (not amplified)		0.167		mV/ °/s
Sensitivity change vs temperature	Delta from 25°C		0.037		%/°C
Zero-rate level(3)			1.23		V
Reference voltage			1.23		V
Zero-rate level change Vs temperature	Delta from 25°C		0.25		°/s/°C
Non linearity	Best fit straight line		±1		% FS
Bandwidth(4)			140		Hz
Rate noise density			0.175		°/s / $\sqrt{\text{Hz}}$
Operating temperature range		-40		+85	°C

表 2.8: LPY5150AL の設定

Measurement range	± 1 ° /s
Sampling rate	50Hz

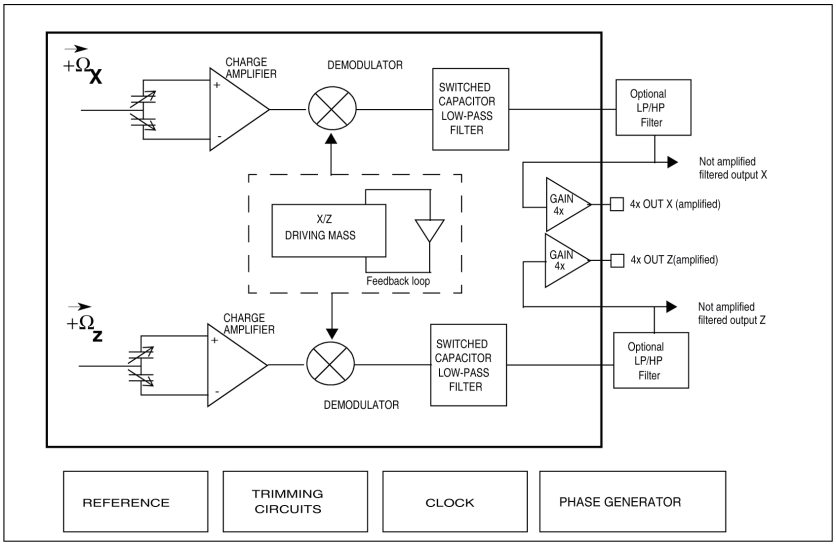


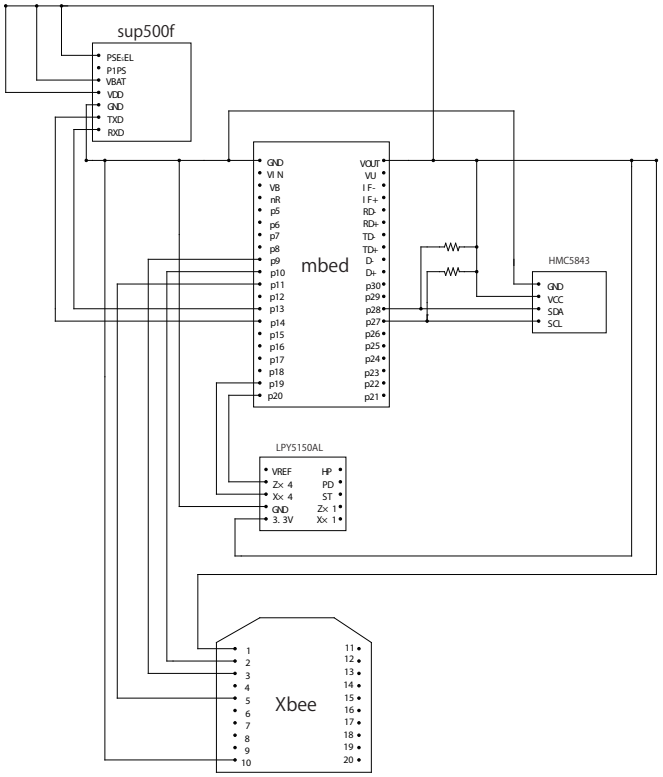
図 2.17: LPY5150AL ブロック図

(出典：http://www.sparkfun.com/datasheets/Sensors/IMU/lpy5150al.pdf)

2011 年 1 月 11 日

2.8　プロトタイプの作成

開発した計測器の回路図を図 2.18 に示す．mbed と GPS モジュール sup500f の通信方式はシリアル通信である．同様に，mbed と Zigbee モジュール Xbee series 2 との通信方式もシリアル通信である．mbed と地磁気センサモジュール HMC5843 の通信方式は SPI 通信である．mbed とジャイロスコープモジュール LPY5150AL の通信方式はアナログ通信である．開発に用いた mbed のソースコードは付録 A に掲載した．



Pin #	Name	Direction	Description
1	VCC	-	Power supply
2	DOUT	Output	UART Data Out
3	DIN / $\overline{\text{CONFIG}}$	Input	UART Data In
4	DIO12	Either	Digital I/O 12
5	$\overline{\text{RESET}}$	Input	Module Reset (reset pulse must be at least 200 ns)
6	PWM0 / RSSI / DIO10	Either	PWM Output 0 / RX Signal Strength Indicator / Digital IO
7	DIO11	Either	Digital I/O 11
8	[reserved]	-	Do not connect
9	$\overline{\text{DTR}}$ / SLEEP_RQ/ DIO8	Either	Pin Sleep Control Line or Digital IO 8
10	GND	-	Ground
11	DIO4	Either	Digital I/O 4
12	$\overline{\text{CTS}}$ / DIO7	Either	Clear-to-Send Flow Control or Digital I/O 7. CTS, if enabled, is an output.
13	ON / $\overline{\text{SLEEP}}$	Output	Module Status Indicator or Digital I/O 9
14	VREF	Input	Not used on this module. For compatibility with other XBee modules, we recommend connecting this pin to a voltage reference if Analog sampling is desired. Otherwise, connect to GND.
15	Associate / DIO5	Either	Associated Indicator, Digital I/O 5
16	$\overline{\text{RTS}}$ / DIO6	Either	Request-to-Send Flow Control, Digital I/O 6. RTS, if enabled, is an input.
17	AD3 / DIO3	Either	Analog Input 3 or Digital I/O 3
18	AD2 / DIO2	Either	Analog Input 2 or Digital I/O 2
19	AD1 / DIO1	Either	Analog Input 1 or Digital I/O 1
20	AD0 / DIO0 / Commissioning Button	Either	Analog Input 0, Digital IO 0, or Commissioning Button

Xbeeeのピン説明

図 2.18: 回路図

2.8.1　センサの装着位置

mbed と各センサモジュールを接続して計測器のプロトタイプを製作した．図 2.19 のように mbed はボールや人との接触が少ないと考えられる背部の中央に装着した．GPS モジュール sup500f と mbed の距離が近いと，衛星と通信できなくなる問題がおこる．それは，mbed のクロックノイズ (96MHz) が原因と考えられる．そこで，mbed と sup500f の距離を離すために sup500f を右肩部に装着した．Zigbee の End Device となる Xbee モジュールアンテナは腰周りに装着すると身体が通信障害となる問題がおこる．そこで，身体が Zigbee 通信の障害とならないように左肩に装着した．ジャイロスコープは図 2.16 に示すように右大腿部に装着した．

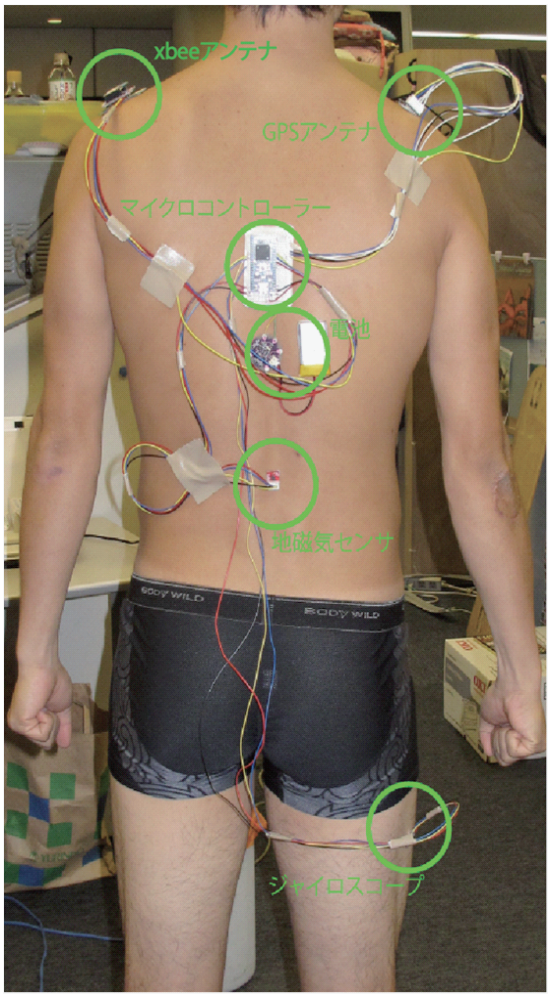


図 2.19: プロトタイプ装着位置

2.8.2　電源

mbed に接続する電源は，3.7V，900mAh のリチウムポリマイオン電池を用いた．図 2.20 にリチウムポリマイオン電池の概要図を示す．

図 2.20: リチウムポリマイオン電池

mbed は 4.5V〜9.0V の電源供給が必要なことから LilyPad リチウムイオンポリマー電池用 DC-DC コンバータをもちいて電圧を 5v に増幅させた．図 2.21 に LilyPad リチウムイオンポリマー電池用 DC-DC コンバータの概要図を示す．

図 2.21: LilyPad リチウムイオンポリマー電池用 DC-DC コンバータ

2.9 計測PC

計測PCとしてのソフトウェアはProcessing言語を用いて開発した．この計測PCはZigbee Coordinatorとシリアル通信を行い，mbedによって処理された各センサからのデータを受信する．計測PCとZigbee Coordinatorの接続にはXbeeエクスプローラを用いた．この計測PCは受信したデータをCSV[7]ファイルとして保存する．同時に，インターネットに接続されている状況であれば，データベースサーバーにそのデータを転送する．データベースサーバーにデータを転送する理由は，後にwebアプリケーションとして分析データを可視化するためである．また，計測PCとmbedをZigbeeを用い通信させることでmbedを制御し，11人分の計測器を時間同期する．さらに，ビデオカメラをIEEE1394ケーブルで計測PCと接続することで，mbedが各センサから獲得した時間のボール状況を，この計測PCが画像情報として保存する．以上のような処理をmbedからデータを受信するごと行うことにより，計測PCでリアルタイムな分析が可能になる．計測PCとZigbee Coordinatorおよびビデオカメラとの接続を図2.22に示す．開発に用いたProcessingのソースコードについては付録Bに掲載した．

[7]Comma Separated Values (データをカンマ (”,”) で区切って並べたファイル形式)

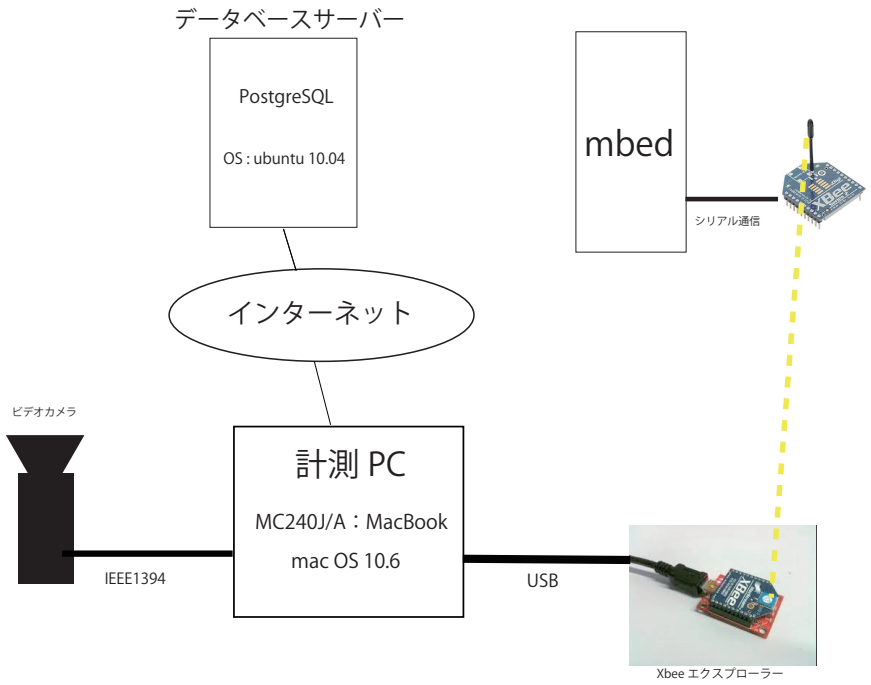


図 2.22: 計測 PC の接続

2.10　まとめ

本章では開発システム概要を示し，開発するためのハードウェアとソフトウェア，開発した計測器のプロトタイプ，計測 PC について述べた．次章では，本章で紹介した計測器のプロトタイプを用いた運動計測について述べる．

第3章　計測

3.1　はじめに

本章では，製作したプロトタイプを用いて，選手が「いつ」,「どこで」,『どの向き』「どのような」動きをしているのかを計測する方法を示すと共に，各センサモジュールの精度について検証する．

3.2　Zigbeeを用いた11人同時計測

本研究における，Zigbeeネットワークの大きな問題は，通信速度と通信距離である．一つのPAN[1]で, 最大で65535個のZigbee端末を接続することが出来るが，トランスミットレート[2]を高くしたり，一度に送信する情報量を多くすると通信できないという問題が発生する．通信距離についてはZigbee Routerを用いてデータを中継することで通信距離を長くすることが可能である．しかし，Routerを用いてデータを中継することにより，通信の時間遅れが発生してしまう．

3.2.1　同時計測と送信データ量の問題

ジャイロスコープセンサのサンプリングレートを50Hzに設定し，Zigbeeのトランスミットレート50Hzでデータを End DeviceからCoordinatorに送信することは，PAN内の一つのEnd DeviceとCoordinatorが通信するだけであれば可能である．しかし，End deviceが11個存在し，それらを同時に通信させることはトランスミットレートが高いため難しい．また，GPS，地磁気センサのサンプリングレートを1Hzに設定し，トランスミットレートを同様に1Hzに設定する．そして，ジャイロセンサの50回分のデータをまとめて送信する

[1]Personal Area Network
[2]1秒間のデータ送信回数

場合，一度に送信するデータ量が多いため 250kbps の上限が問題となり通信できなくなってしまう．このため，50Hz で獲得したジャイロスコープのデータを基に mbed で静止，歩行，走行，ステップワークを運動識別し転送データ量を小さくした．この運動識別については第 4 章で詳しく説明する．mbed で動作識別処理することにより，1 回に送信するデータ型を図 3.1 に示す．そこで，このデータ型を用い，サッカーを想定した 11 個のノードすべてが通信できることを確認する目的の実験 1 を行った．

ID, Counts, direction, WM, pX, pY, Speed

Structure: int , int , int , int , float , float , float \n
1 2 3 4 5 6 7

Example:
8, 1509 , 320 , 1 ,10.3, 22.2, 3.43 \n

Field	name	Example	Description
1	ID	8	Module ID (1 ~ 11)
2	Counts	1509	Data counts
3	Direction	3	Direction(1 ~ 8) 1=0° 2=45° 3=90° 4=135° 9=Error 5=180° 6=225° 7=270° 8=360°
4	WM	1	What is motion(0~3) 0=stoping 1=walking 2=running 3=stepping
5	pX	10.3	Xcordinates of the position (m)
6	pY	22.2	Ycordinates of the position (m)
7	Speed	10.43	Velosity (km/h)

図 3.1: 送信データ型

3.2.2　実験 1

実験 1 ではトランスミットレートを 1Hz とした場合と，2Hz とした場合の 2 つの条件で行った．実験 1 を行うためのハードウェアとして Xbee モジュールを容易に接続できる ArduinoFio を End Device として用い，Xbee エクスプローラーを Coordinator とした．送信するためのデータは図3.1 が示す Example のデータ型 (8,1509,320,1,10.3,22.2,3.43 改行)30byte 分のデータを 1Hz または 2Hz で通信するように設定した．図 3.2 に実験 1 の概要を示す．実験の場所は，慶應義塾大学湘南藤沢キャンパス内のグラウンドで行った．ArduinoFio に接続した Xbee モジュールは被験者の左肩に装着した．被験者はグラウンド内の設定した 100m × 60m のエリア内を 10 分間自由に移動した．被験者は成人 11 人である．ArduinoFio の電源は 3.7v，900mAh のリチウムポリマイオン電池を用いた．

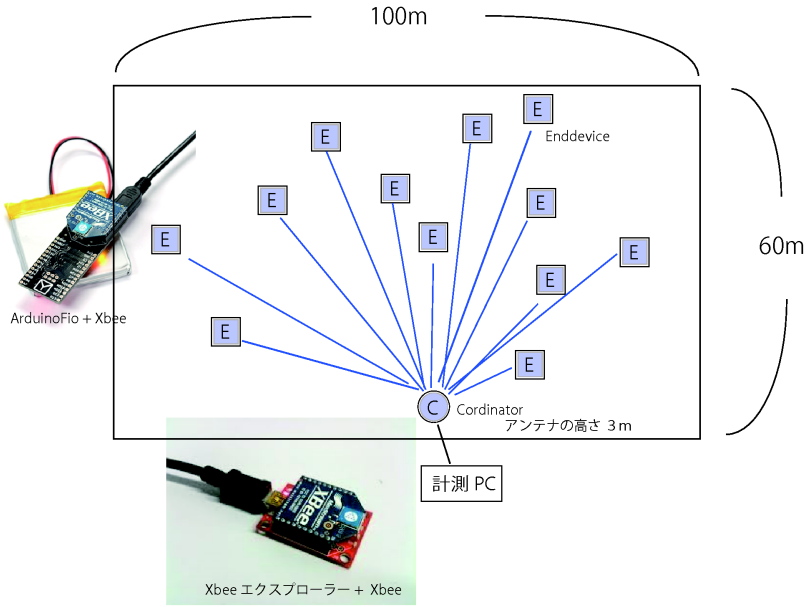


図 3.2: 実験 1 の概要

結果

トランスミットレートが 1Hz の場合，11 人のノードすべてが，安定した 10 分間の通信ができることを確認した．しかし，トランスミットレートが 2Hz の場合はすべてのノードと通信できなくなった．そこで，11 個のノードを 1 つずつ少なくしていき，何個のノードであれば通信できるかを検証した．その結果，4 個以下のノード数であればすべてのノードと通信できることが明らかになった．

3.2.3 通信距離と通信速度の問題

通信距離は Zigbee Router を用いてデータを中継することで通信距離を長くすることが可能である．しかし，Router を用いてデータを中継することにより，通信の時間遅れが発生してしまう．また，通信距離は，Zigbee モジュールに供給される電流の強さ，アンテナの感度，アンテナの位置によって大きく影響する．そこで，Coordinator と一つの End Device の通信距離がどのくらいの距離まで通信可能であるかを知る目的で実験 2 を行った．

3.2.4 実験 2

図 3.3 に示す実験環境で一つの Coordinator と End Device の通信可能距離を測定した．実験 2 では実験 1 と同様の方法で ArduinoFio と Xbee エクスプローラを用いた．End Devic となる ArduinoFio は身長 180cm のサッカー選手の左肩に装着し，Coordinator となる Xbee モジュールを計測 PC に繋ぎ 3 メートルの高さに設定した．End Device が送信するデータは図 3.1 が示す Example のデータ型 (8,1509,320,1,10.3,22.2,3.43 改行)30byte 分のデータを用いた．実験場所は慶應義塾大学 SFC キャンパス内のグラウンドで行った．

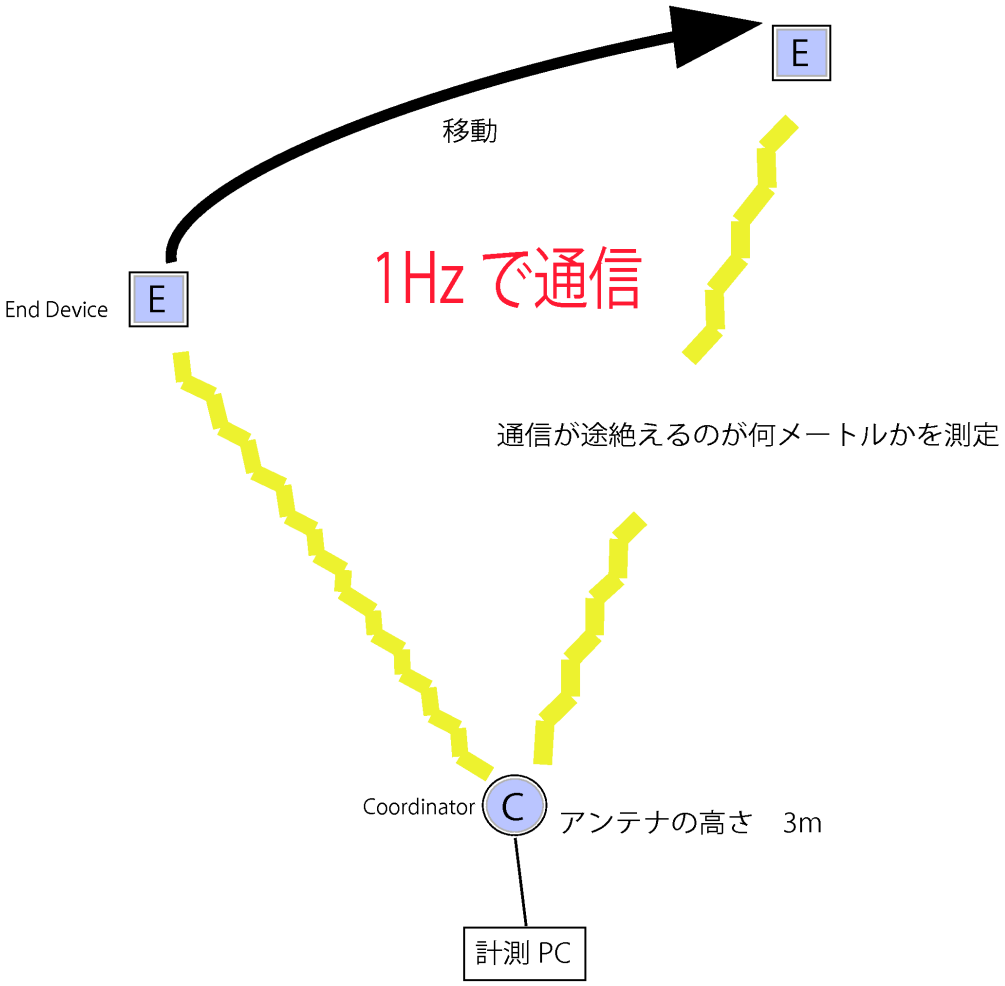

E
移動
1Hz で通信
End Device
E
通信が途絶えるのが何メートルかを測定
Coordinator
C
アンテナの高さ　3m
計測 PC

図 3.3: 実験 2 の概要

結果

通信距離は約 120m 以内の距離であれば安定した通信が 45 分間行えることが計測 PC で確認することができた．

3.2.5 考察

実験 1 の結果から図 3.1 のデータ型であれば 11 人を同時通信する場合，1Hz 以下で通信する必要があると考えられる．そのため，GPS，地磁気センサのサンプリングレートを 1Hz で行わなくてはならない．また，実験 2 の結果から，Router を用いずとも 120m まで通信ができることが確認された．この結果より，Router を設置せずともサッカーグラウンドの範囲内であれば通信が可能であると考えられる．そこで，図 3.4 に示すスター型ネットワークを構築した．また，別の PAN を新たに一つ構築することにより，味方および敵チームとの同時計測も可能となる．

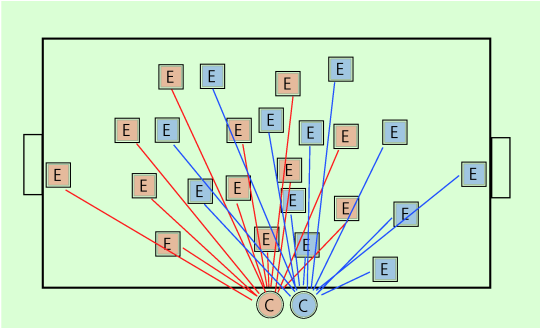


図 3.4: サッカー場に構築する Zigbee ネットワーク

3.3 GPS による位置計測

sup500f から位置情報と速度情報を獲得する．sup500f の仕様書によれば，その位置計測の精度を 2.5m 以内とされている．また，速度計測の精度も時速 0.36km 以内とされている．そこで，静止状態での位置精度を検証する目的の実験 3 と，移動状態での位置精度を検証する目的の実験 4 を行った．sup500f の設定は表 3.1 に示す．GPS からの位置情報は緯度と経度である．この緯度と経度を m(メートル) 単位に変換するために，緯度 1 秒の長さ（緯度 35 度上）

30.8m と経度 1 秒の長さ（緯度 35 度上）25m の値を用いた．実験を行った慶應義塾大学湘南藤沢キャンパスの緯度が約 35 度 20 分であり，20 分の誤差は数 cm にしか満たないことから緯度 35 度上の値を用いた．さらに，サッカー場での絶対座標への変換を行った．図 3.5 が示すようなサッカー場の絶対座標系を設定した．この絶対座標系を設定するために，図 3.6 が示す A0，A1，A2 の位置つまり緯度経度をそれぞれ 1 分間計測した．その平均値をそれぞれの A0，A1，A2 の位置とした．そして

$$\vec{X} = \vec{A1} - \vec{A0} \tag{3.1}$$

の単位ベクトル

$$\vec{e_X} = \frac{\vec{A1} - \vec{A0}}{|\vec{A1} - \vec{A0}|} \tag{3.2}$$

を X 方向の単位ベクトルとし

$$\vec{Y} = \vec{A2} - \vec{A0} \tag{3.3}$$

の単位ベクトル

$$\vec{e_Y} = \frac{\vec{A2} - \vec{A0}}{|\vec{A2} - \vec{A0}|} \tag{3.4}$$

を Y 方向の単位ベクトルとした．図 3.6 に示すサッカー場のある位置 B をサッカー場の絶対座標系で示す場合

$$\vec{b} = \vec{B} - \vec{A0} \tag{3.5}$$

の位置ベクトルと X 方向の単位ベクトルとの内積値

$$\vec{b} \cdot \vec{e_X} \tag{3.6}$$

が位置 B の X 座標値となり，その位置ベクトルと Y 方向の単位ベクトルとの内積値

$$\vec{b} \cdot \vec{e_Y} \tag{3.7}$$

が位置 B の Y 座標値となる．

表 3.1: sup500f の設定

Serial port baud rate	9600
Output message	RMC (NMEA)
Sampling rate	1Hz

図 3.5: 絶対座標系設定

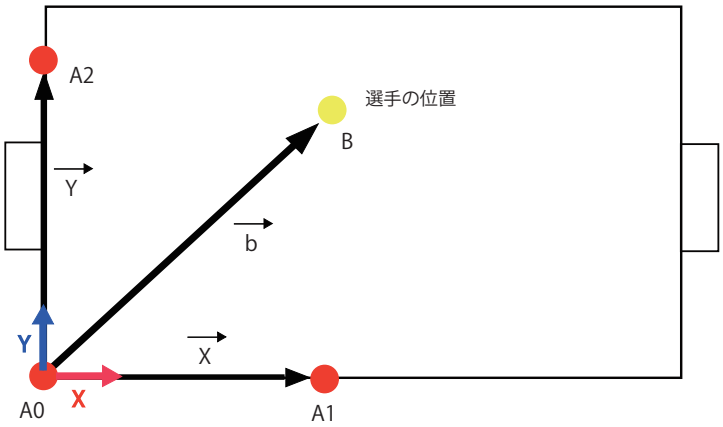


図 3.6: 絶対座標系における位置

3.3.1　実験 3

実験 3 は sup500f を用いた場合，静止状態の計測がどのくらいの精度であるのかを確認するために行った．実験は慶應義塾大学湘南藤沢キャンパス内のグラウンドで行った．そこで，実験 2 と同じ被験者の右肩に sup500f を装着し，静止状態の位置を 10 分間計測した．サンプリングレートは 1Hz で計測した．

結果

図 3.7 に 1 秒ごとに 10 分間計測した位置をプロットしたものを示す．図 3.7 は初期値を (0,0) にし，静止位置計測誤差を示すものである．

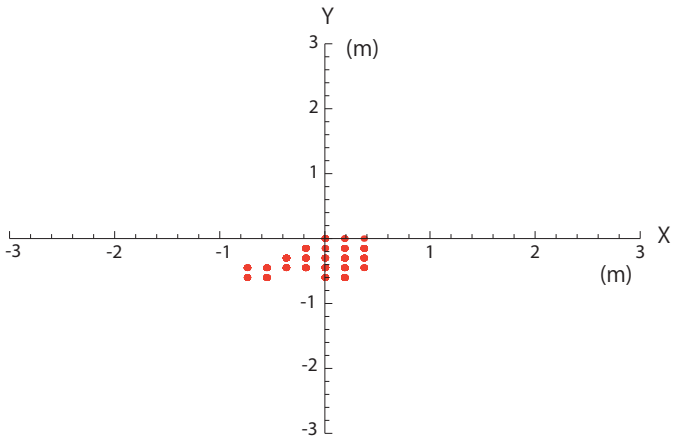


図 3.7: 静止位置誤差

3.3.2　実験 4

実験 4 は sup500f を用い，移動した場合の位置精度を確認するために行った．実験は慶應義塾大学湘南藤沢キャンパス内のグラウンドで行った．被験者は実験 3 と同じ被験者，サンプリングレートは 1Hz で計測した．実験は，歩行，ジョギング程度の低速歩行，中距離走を走るような中速走行，短距離走を全速力で走るような高速走行の 4 試技を行った．この 4 試技はサッカー場に設定し

たある２地点 A，B を直線的に往復する試技である．実験試技の往復回数は表 3.2 に示す．図 3.8 の A，B の位置はそれぞれの位置で静止状態を 1 分間計測し，それぞれを平均としたものである．

表 3.2: 実験試技と往復回数

実験試技	往復回数
歩行	3
低速歩行	3
中速歩行	2
高速歩行	2

結果

図 3.8 に歩きの軌跡と図 3.9 にその速度示す．図 3.10 にジョギング程度の走りの軌跡と図 3.11 にその速度を示す．図 3.12 にジョギングの時より速い速度の走りの軌跡と図 3.13 にその速度を示す．図 3.14 に全力疾走した走りの軌跡と図 3.15 にその速度を示す．

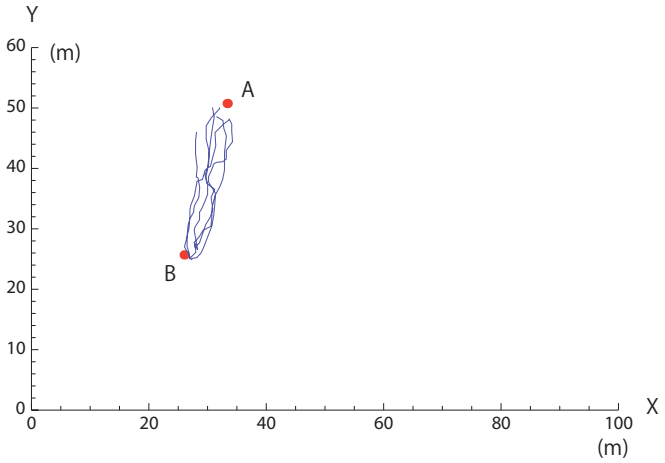


図 3.8: 歩行軌跡

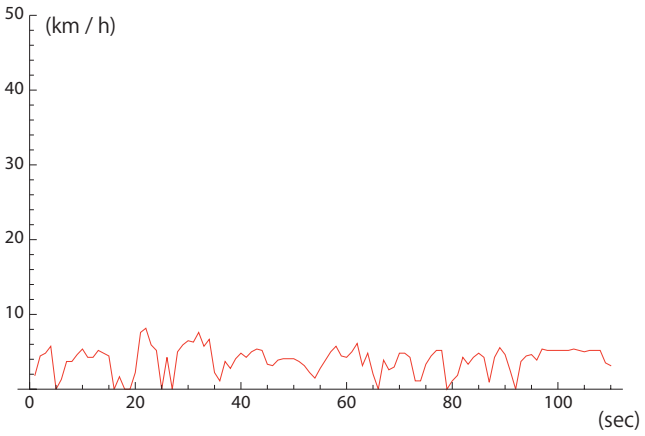


図 3.9: 歩行速度

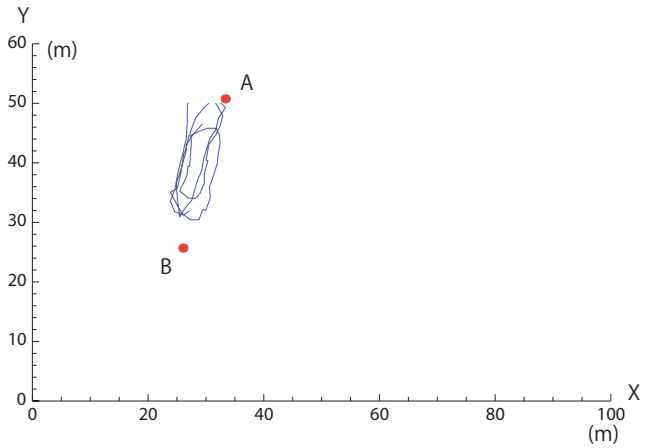


図 3.10: 低速走行往復軌跡

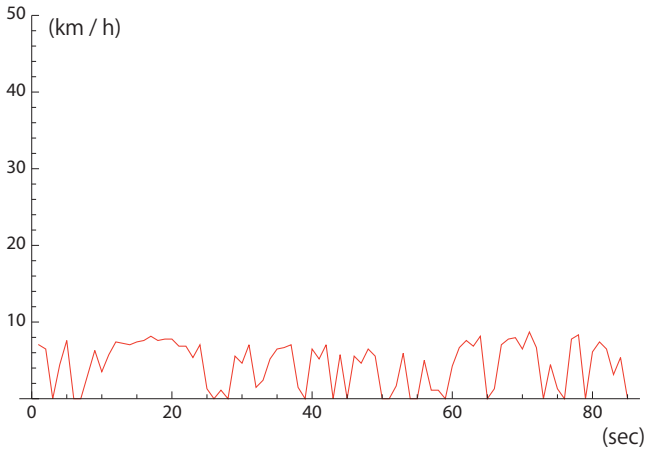


図 3.11: 低速歩行速度

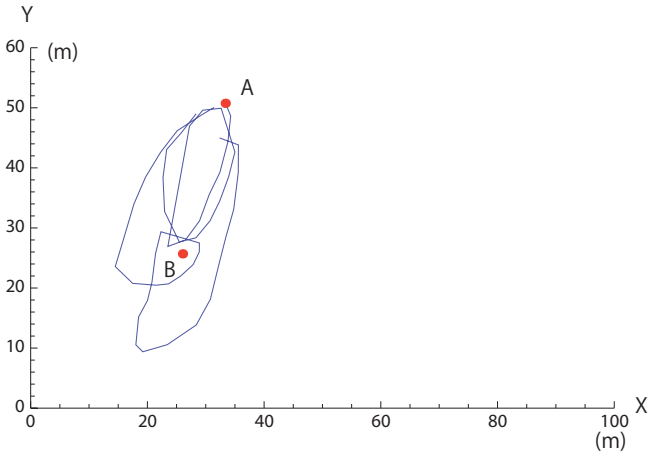


図 3.12: 中速走行往復軌跡

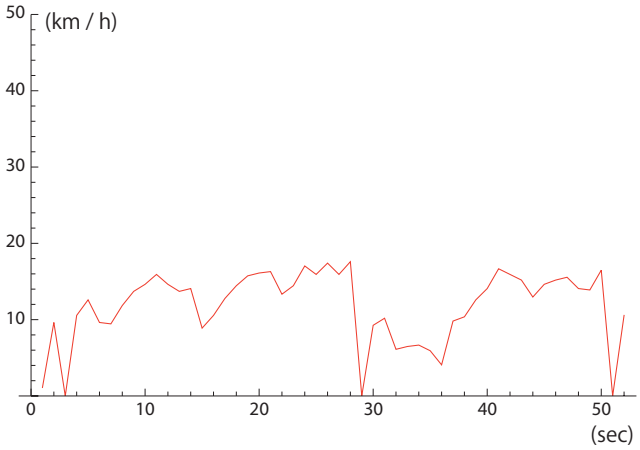


図 3.13: 中速走行速度

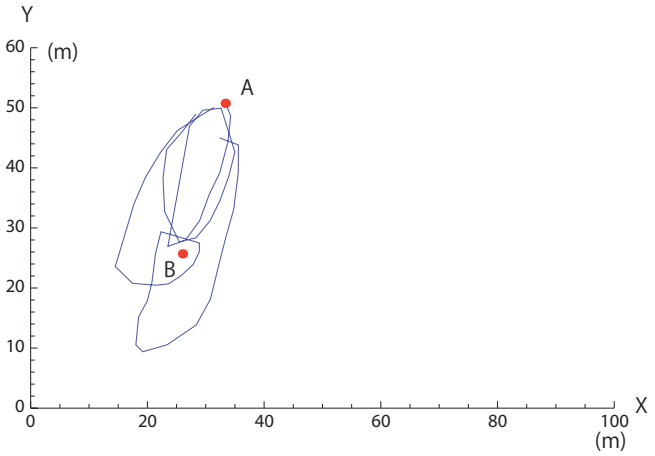


図 3.14: 高速走行往復軌跡

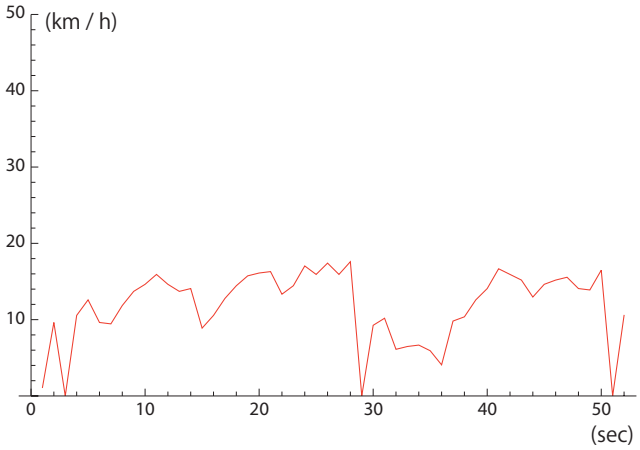


図 3.15: 高速走行速度

3.3.3 考察

実験 3 の結果から，静止状態であるにもかかわらず，半径約 1m 以内の範囲で推定位置が移動してることが確認された．実験 4 の結果からは歩行時の移動であれば，実際に移動した直線上 (図 3.8 の A 点と B 点を結んだ直線上) から約 2.0 メートル以内の位置に移動軌跡を確認できた．しかし，速度が速くなるにつれ，実際に移動した直線上から離れた位置に移動軌跡が確認されるようになった．GPS の誤差は主に，衛星位置誤差，衛星時計誤差，電離層遅延，対流圏遅延，受信機雑音が原因とされている [19].

衛星位置誤差

GPS 衛星が軌道における真位置と測位計算に使われる位置とのずれを衛星位置誤差という．GPS 衛星軌道情報は地上の制御部で監視され，位置予測されたデータが各衛星に送られる．衛星は航法メッセージとして軌道情報を地上に放送しているが，数メートルから十数メートルの誤差を避けることはできない．GPS 衛星軌道情報はエフェメリスデータと呼ばれている．エフェメリスデータは各衛星とも約 2 時間で更新され，更新されてから時間の経過とともに信頼度が低下する．

衛星時計誤差

GPS 衛星には，時刻同期用の正確な発信機としてセシウム原子時計が搭載されている．精度は 10^{-13} 秒程度といわれているが，1 日の間には 10^{-8} 秒程度の時刻誤差を生じる．距離に換算すると約 6m であるため補正を行わなければならない．地上のモニター局の受信機で，時計の同期誤差を連続的に監視し，1 日に 1 度は時計の補正データを各衛星に送信している．

電離層遅延

電離層は，地上 100Km 付近から 1000Km にまで及んでいる大気層である．希薄な大気分子が太陽からの赤外線によって電子とイオンに解離している．質量が小さく外界の電界によって振り回されやすい電子が，電波の伝播，特にス

ピードに大きく影響を与えるため，測定された擬似距離に伝播遅延を引き起し，測位精度を下げる．

対流圏遅延

GPS 信号は乾燥空気および水蒸気から構成される地球大気の下層部分によっても屈折される．そのため，大気圧，温度，湿度，水蒸気分圧などが誤差の要因となる．

受信機雑音

受信機雑音とは，信号に関係なく GPS 帯域において，アンテナで受信される電波による放射をすべて含む．それは，アンテナ，増幅器，ケーブル，受信機からの雑音，マルチアクセル雑音，量子化雑音によるものである．

マルチパス

マルチパスとは，信号が二つ以上の経路を通ってアンテナに到達する現象のことである．マルチパスによる距離測定値誤差は，反射信号の強さと直接信号に対する反射信号の遅延に依存している．

実験 3 の結果については，上記の 5 つの要因が考えられる．しかし，実験 4 の結果については，各実験試技環境が同じであることから，衛星位置誤差と衛星時計誤差が大きな要因であると考えられる．それは，エフェメリスデータ[3]が更新されてから，時間経過とともにエフェメリスデータの信頼度が低下するためであり，同様に衛星時計の信頼度も時間経過とともに低下するからである．実験 4 の試技は，歩行，低速歩行，中速歩行，高速歩行の順番で約 5 分ごとに行った．そのため，歩行試技の実験と，高速歩行試技の実験では約 20 分違うことになり，この時間経過が誤差を増大させたと考えられる．また，速度変化により，その位置誤差の精度が変化したと考えることもできる．しかし，人間の歩行速度から走行速度程度の速度変化が GPS 計測の位置誤差を増幅させるという報告はされておらず，実験 4 だけでは断言することはできない．速度計

[3]GPS 衛星軌道情報

測の精度については，移動速度が低速な場合，誤検出で速度が 0km となっている場合がある．原因としては，位置誤差の問題以外に，サンプリングレートが 1Hz であるため，方向が変わる位置では図 3.16 が示すように，実際の移動距離（赤線）より短い移動距離（青線）と計測されてしまうため，速度が非常に小さい値を示すことも考えられる．そのため，GPS のサンプリングレートを高めることでその精度を改善できると考えられる．ただし，今回開発するシステムでは Zigbee のトランスミットレートが実験 1 より 1kHz 以下にする必要がある．このため，GPS のサンプリングレートを高くするためには，Zigbee での通信の改良も必要である．以上のことから，sup500f を用いた GPS 計測では，その推定位置精度に問題がある．その問題の解決策として，より高性能な GPS モジュールを用いることが考えられる．Cresent OEM board を陸上競技で利用した場合，その位置精度を 50cm 以内に抑えることができることも報告されている [20]．しかし，より高性能な GPS モジュールを用いることにより開発コストが高くなることも考えなければならない．ただし，年々の GPS の精度の向上，GPS モジュールのコスト低下を考えると将来的に安価で高性能な GPS モジュールを使えるようになることが期待できる．GPS の精度は，センサモジュールの改善以外に DGPS[4]や SBAS[5]を用いた補正システムを用いることで年々その精度は改善されてきている．さらに，2010 年 9 月に日本の準天頂衛星として『みちびき』が打ち上げられるなどの，今後の GPS 精度向上に期待することができる．

[4]Differential GPS

[5]Satellite-basedAugmentation System

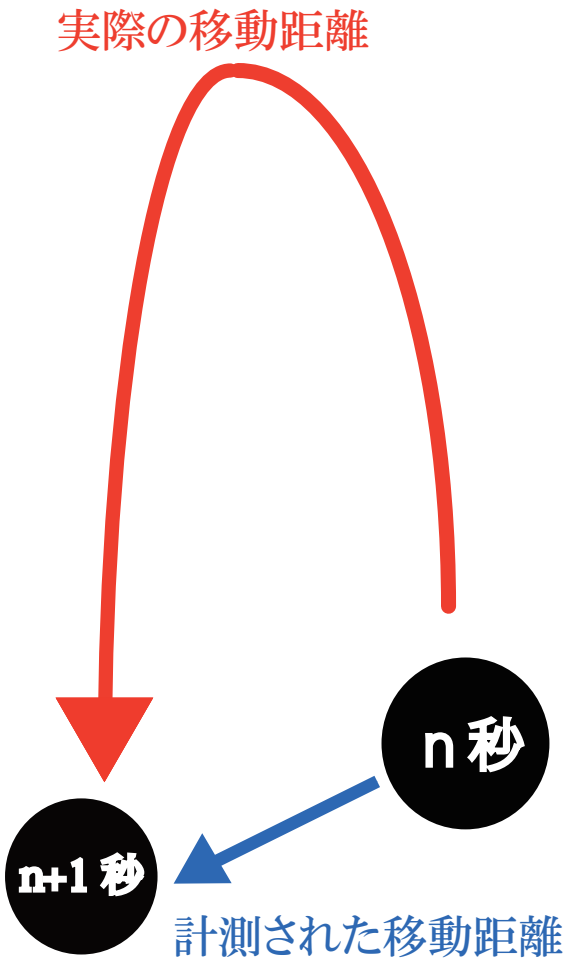


図 3.16: 実際より短い移動距離の計測

3.4　地磁気センサの計測

地磁気センサ HMC5843 を背部に装着し，地磁気センサの座標系を図 3.17 が示すように設定した．X 軸は垂直軸と設定した．Y 軸方向は矢状軸[6]と設定した．Z 軸方向は前額軸[7]と設定した．

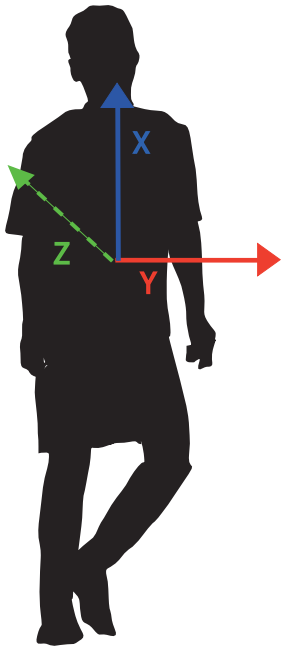


図 3.17: 地磁気の座標系

地磁気は地面方向に対する地場，図 3.17 の座標系の場合 X 軸成分の絶対値が最大になる．そのため，図 3.17 が示す場合の X 軸方面の磁力が 3 つの軸成分の中で最大の時，上半身が地面に対して鉛直方向に向いていると考えられる．もし，Y 軸成分の絶対値が最大になった場合は，上半身の矢状面[8]が地面に対して平行方向になっていると考えられる．また，X 軸成分の絶対値が最大

[6]左右相称な生物の体の正中に対し平行な軸
[7]身体を左右に貫く軸で矢状面に直交する
[8]左右相称な生物の体の正中に対し平行に体を左右に分ける面

になった場合は，前額面[9]が地面に対して平行方向になっていると考えられる．そこで，X 軸方面の磁力が 3 軸成分の中で最大の時，YZ 成分を用いて方位を検出することにする．最大でない場合は，方位推定の処理は行わず Erroer を出力する．3 軸地磁気センサと 3 軸加速度センサの両方を用いることで，地磁気センサの傾きを推定し，その傾きを基に 3 軸成分の地磁気センサから方位を検出することもできるが，その条件として地磁気センサが静止状態になっている必要がある．3 軸加速度センサを用いる方法は，重力成分を検出することで地磁気センサの傾きを推定する方法である．しかし，不規則な運動をしているサッカー選手に 3 軸加速度センサを装着し，重力成分だけを検出することは非常に困難である．そのため，X 軸方面の磁力が 3 軸成分の中で最大になる場合，直立姿勢になっていると仮定し，YZ 成分を用いて方位を検出することにする．そこで，HMC5843 を用いた場合どのくらいの方位を推定できるかを検証する目的で実験 5 を行った．

3.4.1 実験 5

実験 5 では，図 3.18 に示す 0°，45°，90°，135°，180°，225°，270°，315° の 8 つ方向に体の正面を向けそれぞれ 1 分ずつ計測した．実験は慶應義塾大学湘南藤沢キャンパス内のグラウンドで行った．被験者は身長 180cm のサッカー選手で，地磁気センサの座標系が図 3.17 になるよう背部の中心に装着した．サンプリングレートは 1Hz で計測した．

結果

8 方向でそれぞれ検出された Y 軸成分の値 mZ_{nt} と Z 軸成分の値 mZ_{nt}(n は図 3.18 で示す方位の角度，t は時間 (秒) を示す) を用いてベクトル $\vec{m_{nt}}$ を定義した．

$$\vec{m_{nt}} = \begin{pmatrix} mY_{nt} \\ mZ_{nt} \end{pmatrix} \tag{3.8}$$

そして，$\vec{m_{nt}}$ の単位ベクトル

$$e\vec{m}_{nt} = \frac{\vec{m_{nt}}}{|\vec{m_{nt}}|} \tag{3.9}$$

[9]生物の体を腹側と背側に分割する平面

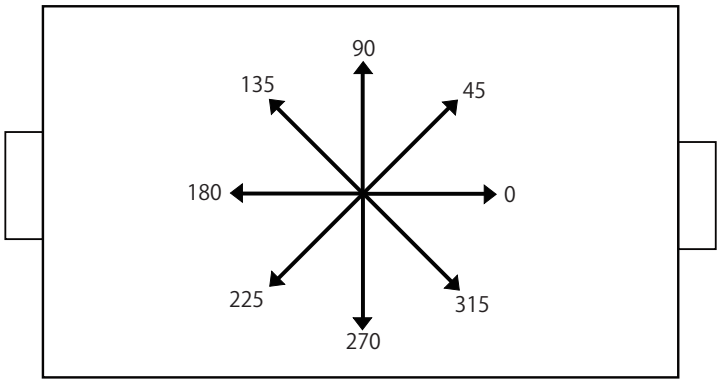


図 3.18: グラウンド場での 8 方位設定

のそれぞれの方位をプロットしたものを図 3.19 に示す.

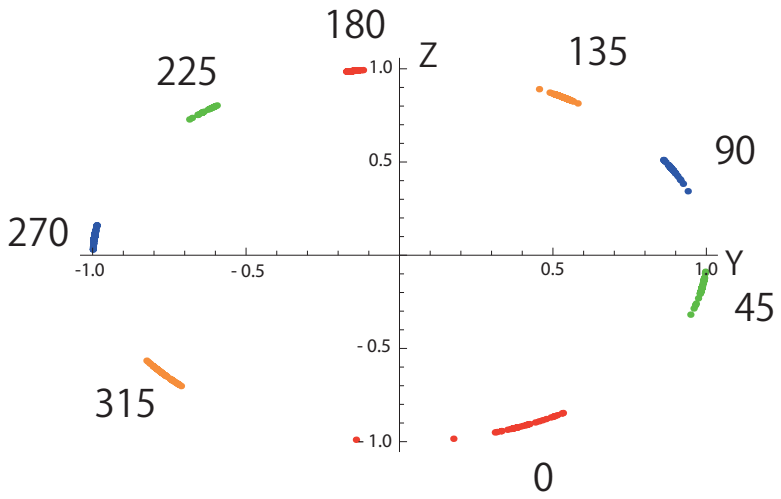


図 3.19: 実験 5 の結果

3.4.2 サッカー場における方位検出方法

実験 5 の結果より，地磁気センサ HMXC5843 の精度であれば 8 方向までの検出は十分可能であると考えられる．8 方向の検出ができれば，前方向に進んでいるのか，後ろ方向に進んでいるのか，それとも横方向に進んでいるのかを分析することができる．そこで，図 3.20 に示すように 8 方向のそれぞれ値の

平均値を 0°，45°，90°，135°，180°，225°，270°，315° の方位と定義した．また定義した方位の単位ベクトルをそれぞれ $e\vec{M}_0$，$e\vec{M}_{45}$，$e\vec{M}_{90}$，$e\vec{M}_{135}$，$e\vec{M}_{180}$，$e\vec{M}_{225}$，$e\vec{M}_{270}$，$e\vec{M}_{315}$ とした．これにより，サッカー場での 8 つの方位を推定する．その方法は，ある地点で計測した Y 軸 Z 軸成分のベクトル $\vec{m}_t(t$ は時間 (秒) を示す) と，定義した 8 つの方位の単位ベクトルをそれぞれ内積し

$$l_0 = \vec{m}_t \cdot e\vec{M}_0 \tag{3.10}$$

$$l_{45} = \vec{m}_t \cdot e\vec{M}_{45} \tag{3.11}$$

$$l_{90} = \vec{m}_t \cdot e\vec{M}_{90} \tag{3.12}$$

$$l_{135} = \vec{m}_t \cdot e\vec{M}_{135} \tag{3.13}$$

$$l_{180} = \vec{m}_t \cdot e\vec{M}_{180} \tag{3.14}$$

$$l_{225} = \vec{m}_t \cdot e\vec{M}_{225} \tag{3.15}$$

$$l_{270} = \vec{m}_t \cdot e\vec{M}_{270} \tag{3.16}$$

$$l_{315} = \vec{m}_t \cdot e\vec{M}_{315} \tag{3.17}$$

以上の内積値 l_0〜k_{315} が 8 つの中で最大になった場合の角度を方位とする方法である．

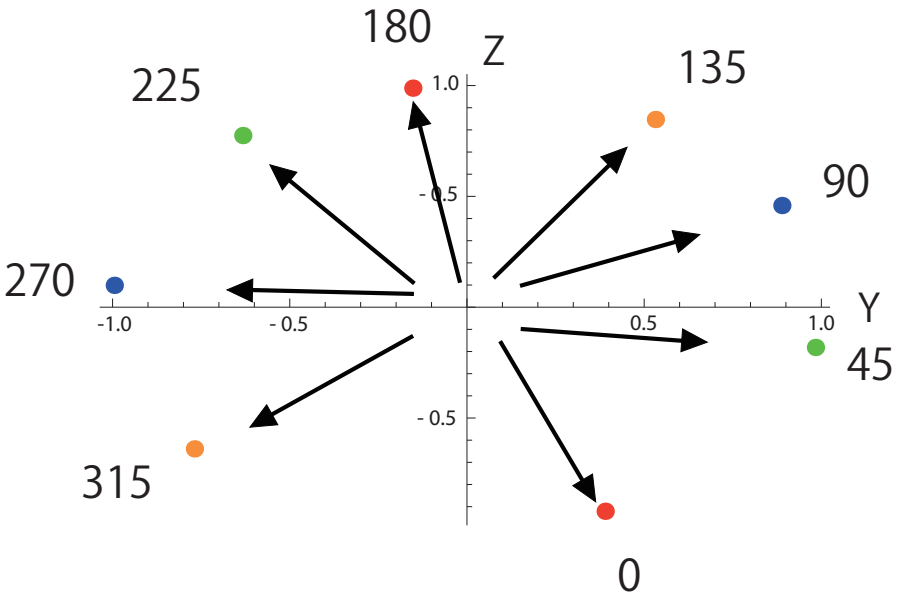


図 3.20: 8 方位の単位ベクトルを定義

3.4.3　考察

実験5の結果から，HMC5843を用いれば8方位の推定ができることが明らかになった．ある選手が歩いている状態を，GPSを用いて位置計測を行うことで図3.21のように歩いた軌跡を推定することができる．しかし，これだけの情報ではこの選手がどのように歩いていたのかは分析できない．後ろ向きで歩いている場合も考えられる．そこで，地磁気センサを用いて方位を算出した図3.21に重ね合わせたものを図3.22に示す．これによりこの選手がどのような向きで歩いていたかということが分析できるようになる．このように，方位センサを用いることでより詳細な運動分析が可能になると考えられる．

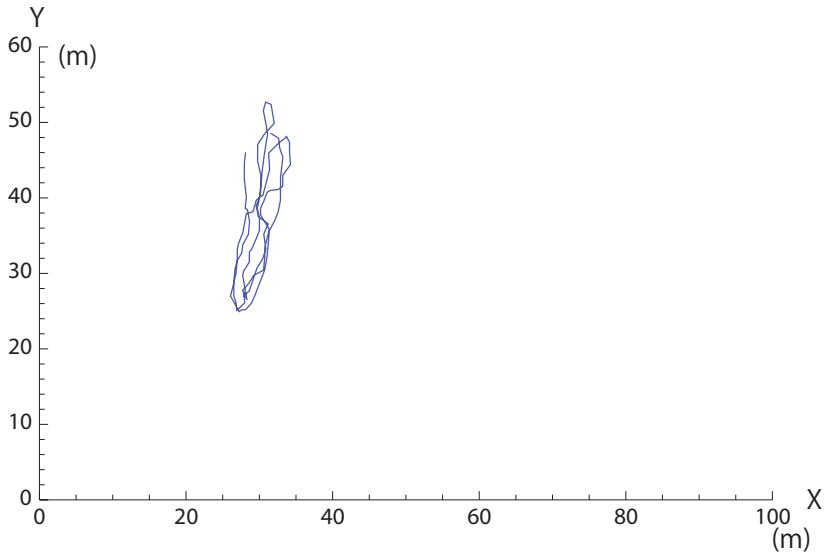


図 3.21: 歩行軌跡の推定

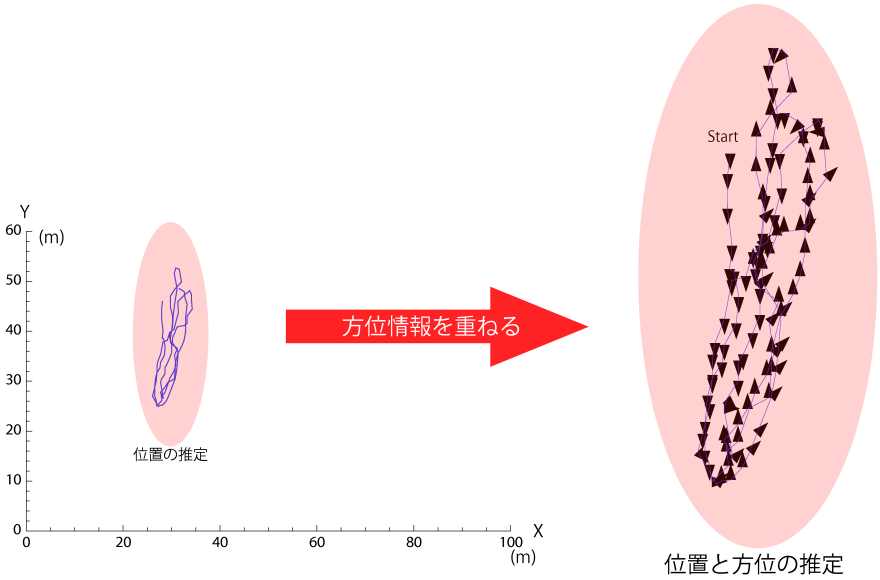


図 3.22: 歩行移動軌跡と方位の推定

3.5　ジャイロスコープを用いた角速度計測

人間の歩く，走る，スッテプワークの運動を大腿部の回転運動を観察すると，それぞれの屈曲伸展の回転速度と，内転外転の回転速度が異なることが分かる．そこで，サッカー選手の右大腿にジャイロスコープを装着し，歩行，走行，ステップワークの運動中の大腿部屈曲伸展角速度と大腿部内外転角速度を計測する実験 6 を行った．大腿部の回転座標系は図 3.23 に示すように設定した．

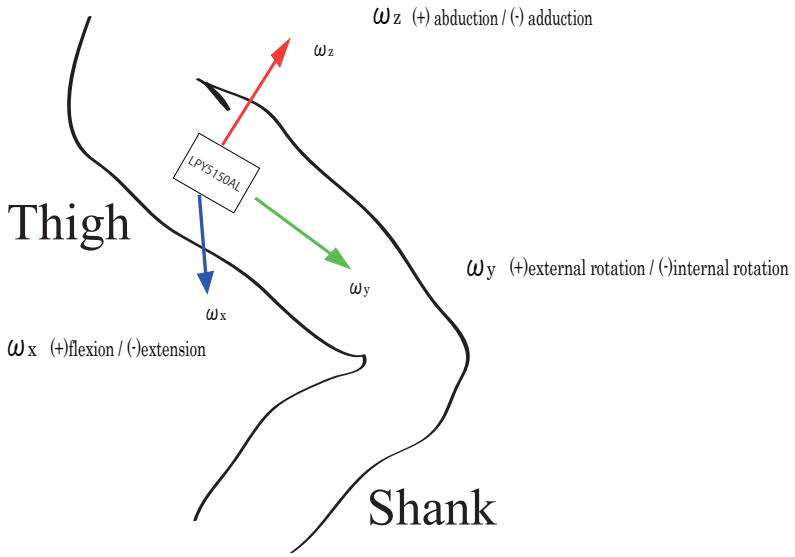


図 3.23: 大腿部の回転座標系

3.5.1　実験 6

実験 6 は，3 人の被験者にジャイロスコープ LPY5150AL を右脚大腿部に装着し，3 試技の実験を行った．実験試技は実際のサッカーの試合中を想定した歩行，走行，ステップワークである．それぞれの試技で，被験者はグランド内を自由に移動した．被験者は日本サッカー協会が指定する地域リーグに所属する 3 人のサッカー選手である．表 3.3 に被験者データを示す．実験場所は慶應義塾大学湘南藤沢キャンパス内のグラウンドで行った．ジャイロスコープ LPY5150AL の計測はサンプリングレート 50Hz で行った．

表 3.3: 被験者データ

	sex	height	weight
Subject A	M	165cm	60kg
Subject B	M	179cm	68kg
Subject C	M	179cm	72kg

結果

3 人の選手の歩行中の大腿部屈曲伸展角速度と大腿部内外転角速度を 2 秒間を切り出したものを図 3.24 に示す．同様に，走行中の大腿部屈曲伸展角速度と大腿部内外転角速度を図 3.25 に，ステップワーク中の大腿部屈曲伸展角速度と大腿部内外転角速度を図 3.26 に示す．

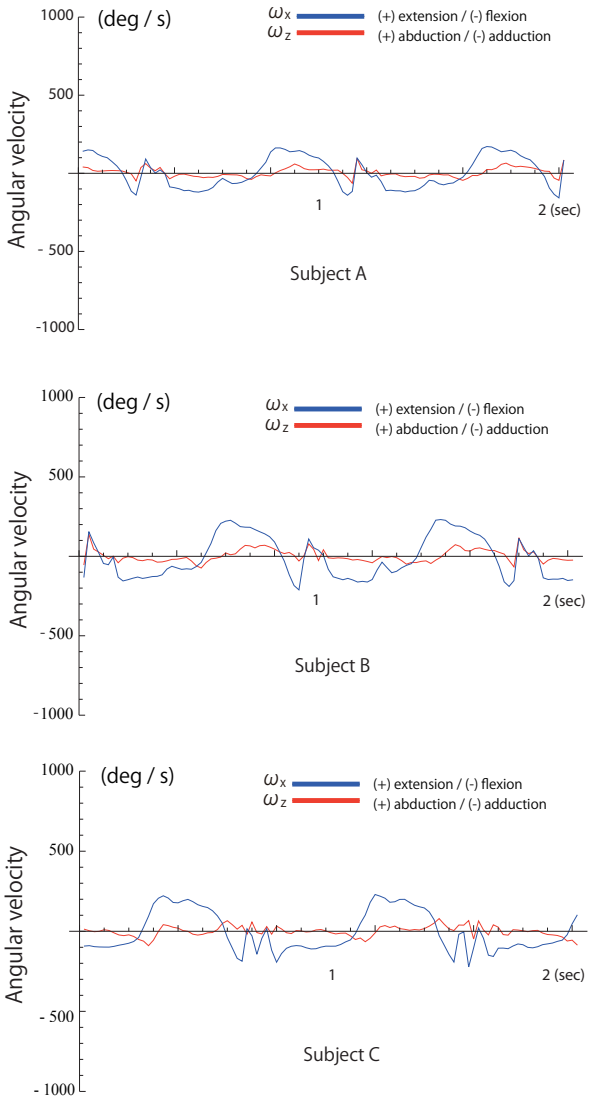


図 3.24: 走行中の大腿部屈曲伸展角速度と大腿部内外転角速度

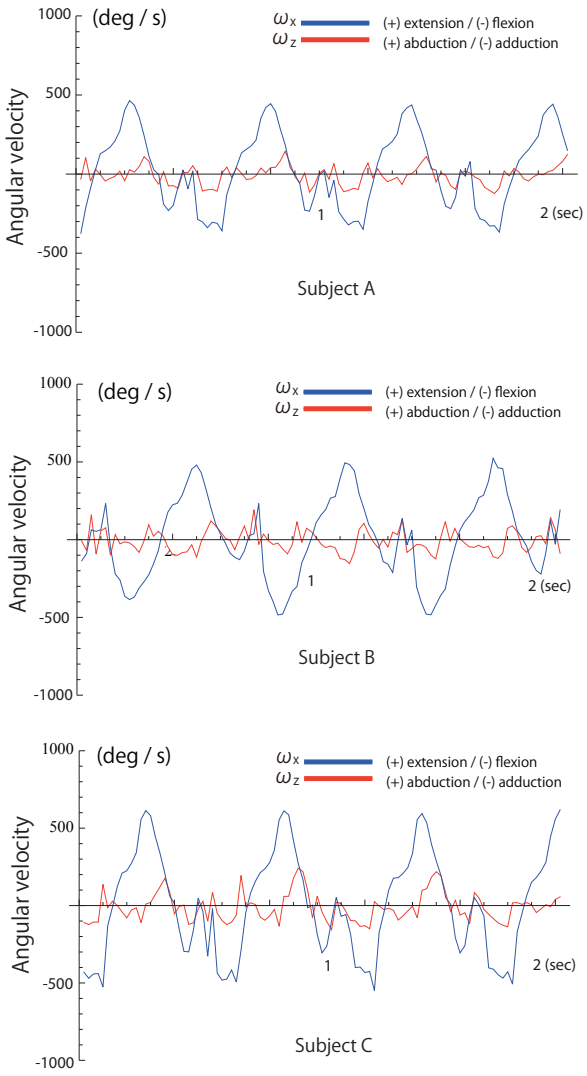


図 3.25: 走行中の大腿部屈曲伸展角速度と大腿部内外転角速度

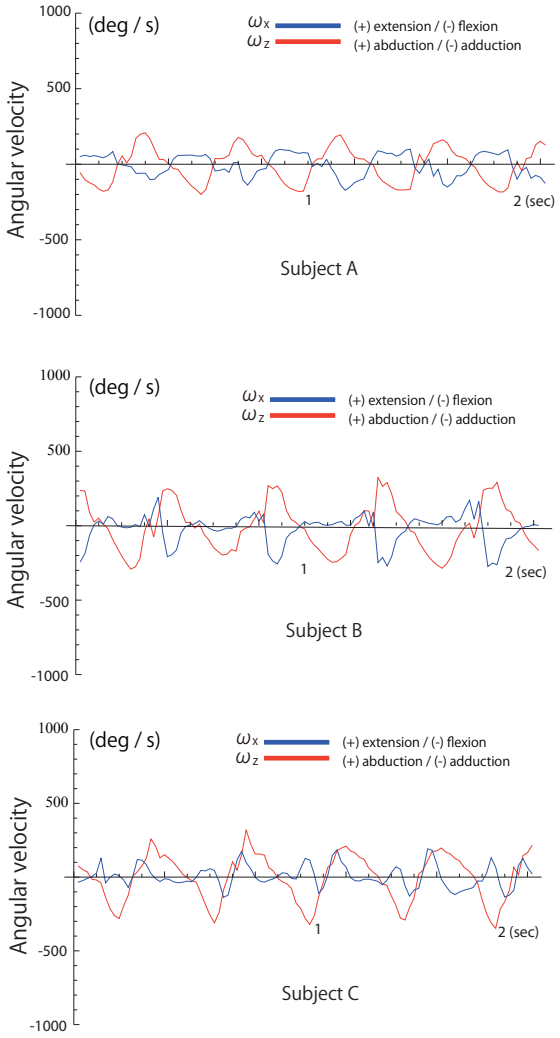


図 3.26: ステップワーク中の大腿部屈曲伸展角速度と大腿部内外転角速度

3.5.2　考察

実験 6 の結果から，被験者 3 人の歩き，走り，ステップワークの運動中，大腿部内外転角速度と屈曲伸展角速度にそれぞれ特徴があることが明らかになった．歩行と走行を比較した場合，走行中の方が屈曲伸展回転角速度の極値の絶対値が大きくなった．また，屈曲伸展の周期が走行時の方が速くなった．スッテプワークでは，内外転角速度の極値の絶対値が歩行，走行中と比べて大きくなった．対照的に，屈曲伸展角速度は小さくなった．そこで，歩行，走行，ステップワーク中の大腿部内外転角速度と屈曲伸展角速度の特徴に着目し，歩行，走行，ステップワークの運動識別を行うことにした．この運動識別については次章の第 4 章で詳しく説明する．

3.6　まとめ

本章では，Zigbee ネットワークを用いた選手全員の同時計測，GPS による位置計測，地磁気センサを用いた方位計測，ジャイロスコープを用いた大腿部屈曲伸展角度と内外転角速度計測について述べた．オフザボールの選手の分析とは，選手が『いつ』,『どこで』,『どの向きで』,『なにを』しているのかを分析することである．Zigbee を用いることで，敵味方含め 22 人全員の動きを同時計測することができる．つまり,『いつ』の分析を可能にする．GPS を用いた位置計測をすることで『どこで』の分析を可能となった．しかし，GPS モジュール sup500f を用いた場合，その位置精度に問題があることが明らかになった．そのため，現段階ではより高性能な GPS モジュールを用いる対策が必要である．地磁気センサ HMC5843 を用いることで 8 方位の推定が可能になることが明らかになった．つまり,『どの向きで』の分析を可能となった．オフザボール中の『なにを』を分析することは，歩行，走行，ステップワークの分析をすることと同義である．その分析を行うためにジャイロスコープを大腿部に装着し，大腿部の内外転角速度と屈曲伸展回転速度に着目した．そこで次章は，このジャイロスコープから獲得した大腿部の屈曲伸展回転速度と内外転角速度を用いて歩行，走行，ステップワークの運動識別について述べる．

第4章　運動識別

4.1　はじめに

第3章で，Xbeeを用いて11人選手同時計測するためには，各センサモジュールから獲得するデータ量を小さくする必要があると述べた．特に，ジャイロスコープからの信号はサンプリングレート50Hzで獲得されるため1秒間のデータ量が非常に多くなる問題がある．この問題は，11人すべてのZigbee End DeviceがCoordinatorと通信する際の障害となる．そこで，50Hzで獲得したジャイロスコープの信号を用いて，1秒間隔で歩行，走行，ステップワークの運動を自動的に制御コンピュータで行うことにする．この処理を行うことで，End Deviceが1秒間隔でCoordinatorと通信するデータ量を図3.1が示すデータ型まで小さくすることができる．そこで本章は，この運動識別を行うために機械学習であるニューラルネットワークを用いて，mbedが運動識別するための最適なパラメータ値を獲得する方法を示す．

4.2　ニューラルネットワーク

運動識別のための最適なパラメータ値を獲得するために，機械学習の方法として知られているニューラルネットワークを採用した．ニューラルネットは, 生体のニューロンを模擬する人工の素子を用いて構成される. 実際のニューロンの振る舞いは極めて複雑であるため, その動作を単純な式で表すことはできない. ニューラルネットに使われる人工の素子は, 生体のニューロンを厳密に模倣したものではなく, その特定の機能を抽出し, 単純化した工学的モデルである. この工学モデルは大きく２つに分けることができる．それは2値モデルとアナログ値モデルである．2値モデルでは，ニューロン素子がある入力に対して興奮するかしないかということをニューロン素子の出力として解釈するため，ニューロン素子は外部へ２通りの作用しか及ぼさない. この場合，作用が

あるかないかの２通りしかなく中間作用を及ぼし得ないため, ニューロン素子の動作は全か無かの法則に従うことになる．一方，ニューロン素子が毎秒何回興奮するかという観点から, ニューロン素子の出力を解釈すると, それはアナログ値となる．この 1 秒間に興奮した回数を s とし sigmoid 関数 (式 4.1) を用いると, 出力値 y(式 4.2) は 0〜1 の間の値をとるアナログ出力となる．α はゲイン値，θ は閾値である．ゲイン α を大きくするとシグモイド関数はステップ型に近づき，出力は $[1, 0]$ の 2 値のいずれかになる．本研究ではアナログ値モデルを用いることにする．アナログ値モデルでは，図 4.2 が示すように，入力素子 $x_1 x_2 \cdots x_n$ がそれぞれのパラメータ値 (結合重み) $\omega_1 \omega_2 \cdots \omega_n$ との積により素子 S に入力され，素子 S は入力値を総和し，sigmoid 関数 (式 4.1) を用いて，出力値 y(式 4.2) を出力する．

$$sigmoid(s-\theta) = \frac{1}{1+e^{-\alpha(s-\theta)}} \tag{4.1}$$

$$y = sigmoid(s-\theta) \tag{4.2}$$

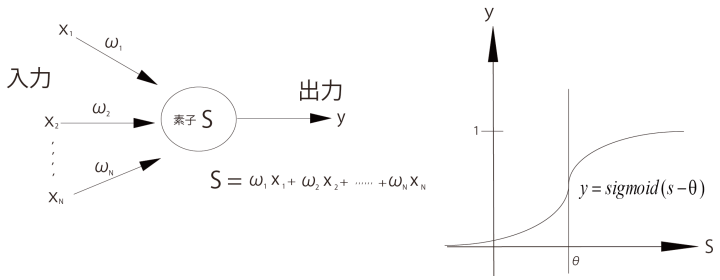


図 4.1: ニューラルネットワークのアナログ値出力モデル

4.2.1 フィードフォワード型ニューラルネットワーク

本研究では，ニューラルネットワークの型はフィードフォワード型を用いる．フィードフォワード型は信号が入力側から出力側に向けて一方向に流れていく

ように構成したニューラルネットワークである．この場合, ネットワークの入力側の素子の値を指定すると, そこから出力側に向けて素子の値は次々と決まることになる. 図 4.2 にフィードフォワード型のモデルを示す.

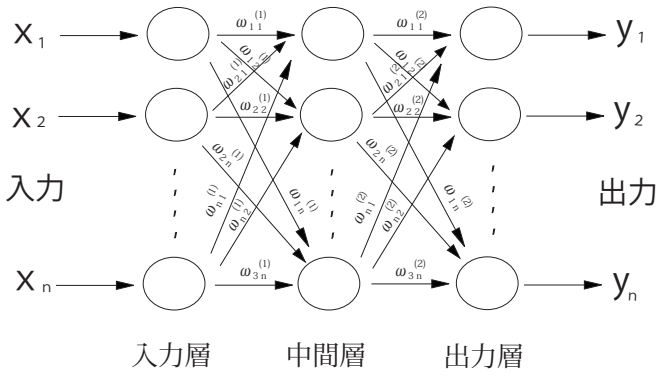


図 4.2: フィードフォワード型ニューラルネットワーク

4.2.2 学習方法

ニューラルネットワークの学習方法としてバックプロパゲーションを用いる．このバックプロパーゲーションを用いることで, 運動識別のための最適な出力をするパラメーター ω(重み) を求めることができる．フィードフォワード型ニューラルネットは，構成される素子の結合重み $\{\omega_{ij}\}$ と閾値 $\{\theta_i\}$ の値を変えることにより，入出力関係を変化させる．そこで．図 4.3 が示すようにバックプロパゲーションは指定された入出力関係を実現するようにネットワーク内部のパラメーター $\{\omega_{ij}\}$ と閾値 $\{\theta_i\}$ を最適な値に定める働きをする.

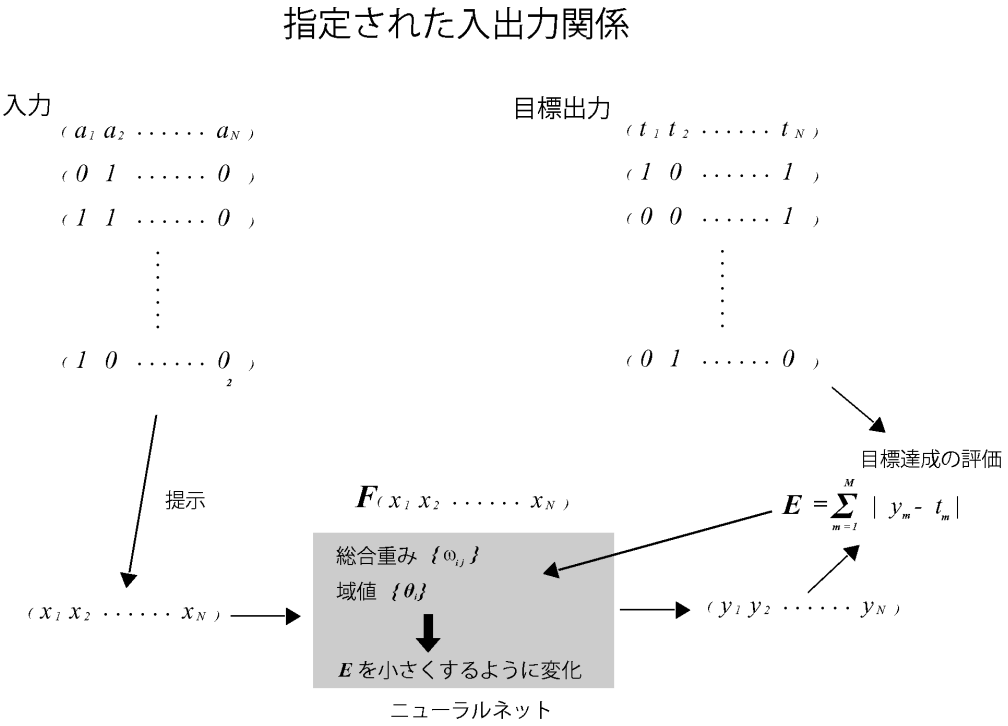


図 4.3: バックプロパゲーションの枠組み

4.3　最適パラメータ獲得方法

フィードフォワード型ニューラルネットワークとバックプロパゲーションを用いて，1 秒ごとに歩行，走行，ステップワークを識別するための最適なパラメータを獲得する目的で 2 つの機械学習を行った．その 2 つの機械学習を，学習 1 と学習 2 とする．学習 1 は，入力値にジャイロスコープから 50Hz で獲得した信号の 1 秒間の平均と分散値を用いる学習である．学習 2 では，ジャイロスコープから 50Hz で獲得した 1 秒間の信号に離散フーリエ変換の処理を行い，パワースペクトル分析から入力値を設定する学習を行った．

4.3.1　学習 1

入力値

実験 6 より, 歩く, 走る, ステップワークの運動の違いから, 大腿の屈曲伸展, 内転外転の振幅の大きさにそれぞれ違いがあることが明らかとなった．そこで，屈曲伸展の信号を a，内転外転の信号を b とした．そして，a，b それぞれの信号の絶対値から 1 秒間の平均値および分散値を算出し，それらを常用対数で表したものを入力値とした．

$$x'_{1t} = \frac{1}{k}\sum_{n=1}^{k} |a_{k(t-1)+n}| \tag{4.3}$$

$$x_{1t} = \log_{10} x'_{1t} \tag{4.4}$$

$$x_{2t} = \log_{10} \frac{1}{k}\sum_{n=1}^{k} (x'_{1t} - |a_{k(t-1)+n}|)^2 \tag{4.5}$$

$$x'_{3t} = \frac{1}{k}\sum_{n=1}^{k} |b_{k(t-1)+n}| \tag{4.6}$$

$$x_{3t} = \log_{10} x'_{3t} \tag{4.7}$$

$$x_{4t} = \log_{10} \frac{1}{k}\sum_{n=1}^{k} (x'_{3t} - |b_{k(t-1)+n}|)^2 \tag{4.8}$$

t(自然数) は t 秒目の入力値であることを示す．この 1 秒間の窓を重ねることなくずらしていくことでサンプルデータ k 個分を 1 単位時間データとしてニューラルネットワークの入力層に与えた．

学習設定

バックプロパゲーションを用いた学習をするために, 実験 6 の被験者である 3 人のサッカー選手の歩き, 走り, ステップワークのデータを 50 ずつ合計 $50 \times 4 \times 3 = 600$ のデータを用いた．学習時の素子数と中間層の数の設定を表 4.1 に示す．フィードフォワード型ニューラルネット構成は図 4.4 に示す. 歩行, 走行, ステップワークの目標出力値を表 4.5 に示す．以上の設定で学習を開始し，20000 回の学習を繰り返した結果，平均誤差が 0.013, 最大誤差が 0.412 近辺の値で収束したため学習を終了させた．図 4.4 に示すフィードフォワード型ニューラルネットワークを設定した理由は，試行錯誤，素子数を入れ替え学習させ，学習が収束したときの平均誤差と最大誤差が一番小さくなったためである．

表 4.1: 学習 1 の設定

学習回数	20000
入力素子数	3
中間層数	2
中間層の素子数	12
出力素子数	3
ゲイン	0.2
学習データ数	600

表 4.2: 目標出力値

	y_1	y_2	y_3
walk	0	1	1
run	1	1	0
step	1	0	1

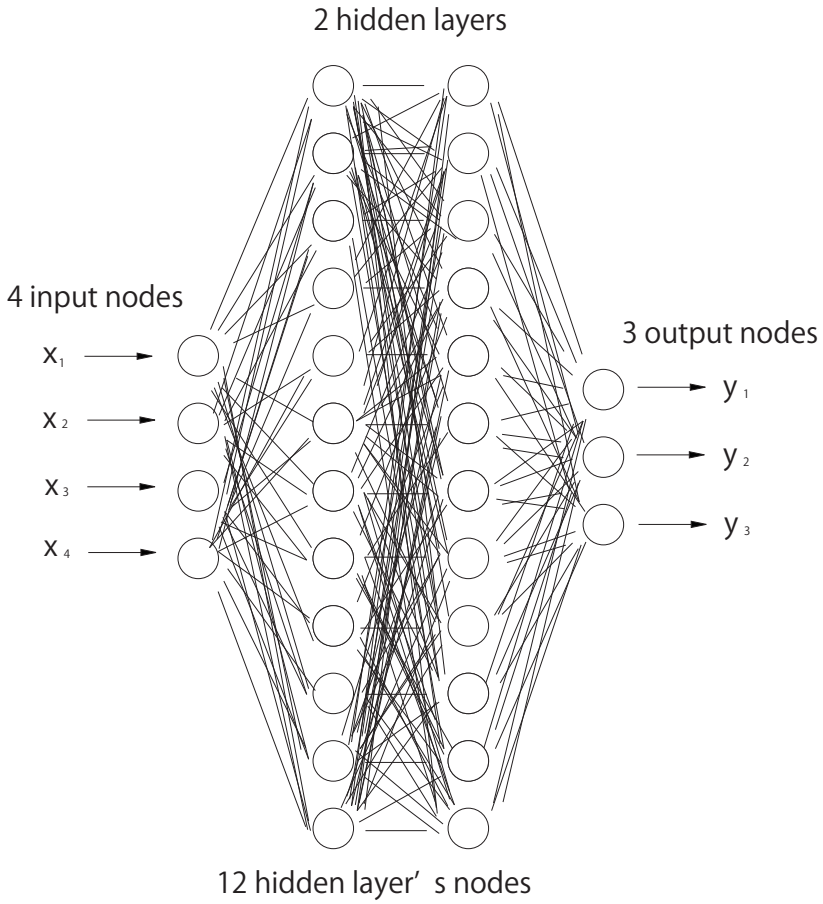


図 4.4: 学習 1 のニューラルネットワーク構成

検証方法

学習結果を検証するために, 学習データとは異なる同じ数の検証データを実験 6 の被験者である 3 人のサッカー選手の歩き，走り，ステップワークのデータから用意した．入力値からニューラルネットワークを通じての出力される値は 0〜1 の間をとるアナログ出力である．そこで出力値を，0.5 以上であれば 1 を， 0.5 未満であれば 0 を出力するようにした. そして，それぞれデータからの出力値が学習結果の検証を表 4.5 の歩き，走り，ステップワークと一致するかを比較し, その正答率を算出した.

検証結果

学習 1 の検証結果を表 4.3 に示す.

表 4.3: 学習 1 の検証結果

movements	percentage of correct answers
walk	94%
run	92%
step	92%

4.3.2 学習 2

キャリブレーションの問題

学習 1 の結果から，平均と分散を入力値とすることにより 90%以上の精度で歩き，走り，ステップワークを識別できた．しかし，大きな問題がある．それは，アナログ値として電圧を出力する LPY5150AL を用いて平均，分散値を用いる場合，計測するごとにキャリブレーション[1]の必要性があることである．それは，アナログセンサモジュールの場合，計測環境の気温や湿度によって，出力される電圧が増幅したり減幅するためである．そこで，この問題を解決するために，50Hz で獲得したジャイロスコープの信号に 1 秒間隔の矩形窓を用いて離散フーリエ変換処理を行い，パワースペクトル分析からの入力値を設定

[1]計測器具の偏りを基準量によって正すこと

する学習 2 を行った．パワースペクトルは，信号のある周期帯における振幅の強さを示す．そのため学習 2 の方法を用いることにより，各ジャイロスコープのキャリブレーションを行う必要性がなくなる．

パワースペクトルの特徴

実験 6 の被験者の歩き，走り，ステップワークの運動の 50 秒分の信号に 1 秒間隔の矩形窓関数を用いて離散フーリエ変換処理を行い，1〜8Hz までのパワースペクトルをそれぞれ重ねたものを以下の図に示す．このパワースペクトルの分析では，離散フーリエ変換の計算量を軽減する目的と人間の運動で大腿部の屈曲伸展角速度と，内転外転角速度が 1 秒間に 8 周期分以上検出されることは考えられないことから，1Hz から 8Hz までの周波数帯を分析対象とした．短形窓を用いた理由は，8Hz 以上の高周波成分を分析対象としないためである．図 4.5 と図 4.6 より，歩きの外転内転，屈曲伸展ではパワースペクトルの大きさの最大が 1Hz に集中することが分かる．図 4.7 と図 4.8 より，走りの場合は内転外転のパワースペクトルの最大は 2〜5Hz の間で分散しているが，屈曲伸展の場合は 2Hz に集中している．図 4.9 と図 4.10 より，ステップワークの場合，内転外転のパワースペクトルの最大が 2Hz に集中するが，屈曲伸展の場合，歩行時と走行時のパワースパクトルを比較すると，パワースペクトルが最大となる周波数帯が分散している．

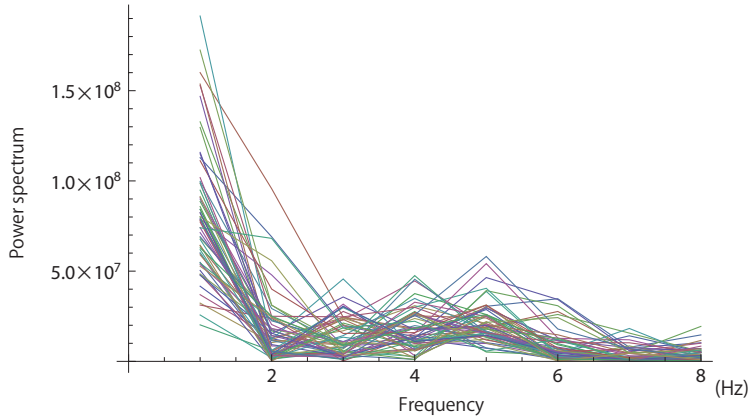


図 4.5: 1 秒間歩行における内転外転回転角速度のパワースペクトル

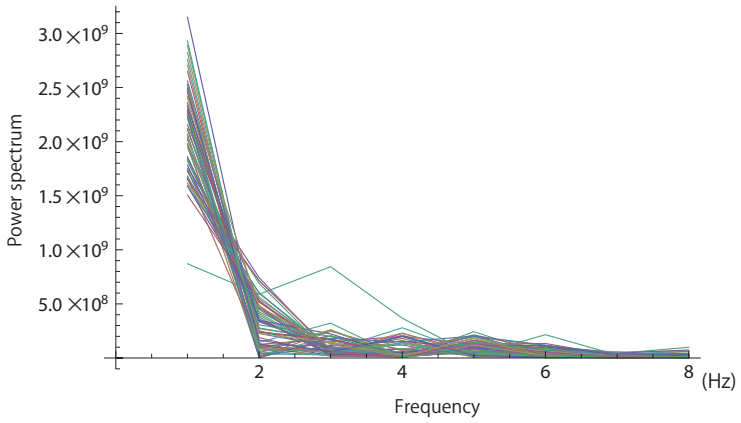


図 4.6: 1 秒間歩行における屈曲伸展回転角速度のパワースペクトル

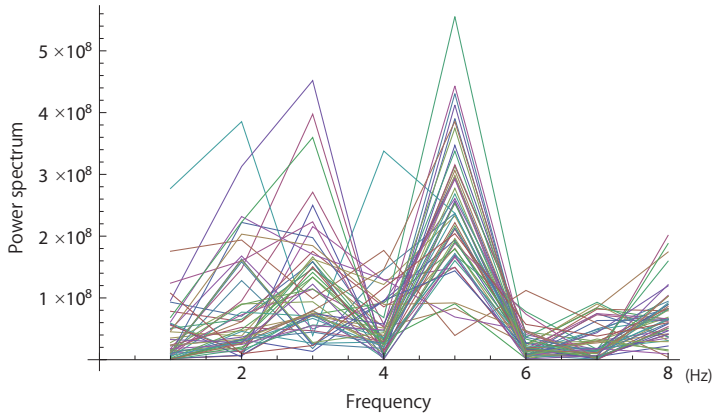


図 4.7: 1 秒間走行における内転外転回転角速度のパワースペクトル

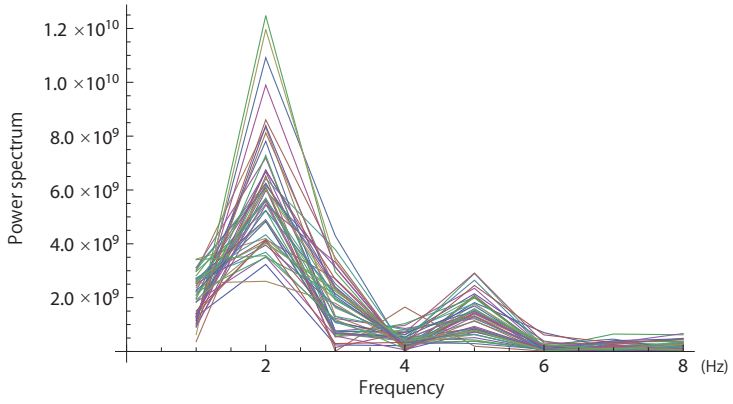


図 4.8: 1 秒間走行における屈曲伸展回転角速度のパワースペクトル

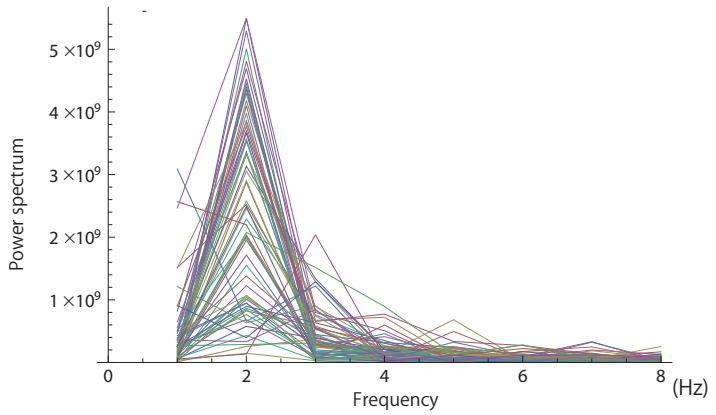


図 4.9: 1 秒間ステップワークにおける内転外転回転角速度のパワースペクトル

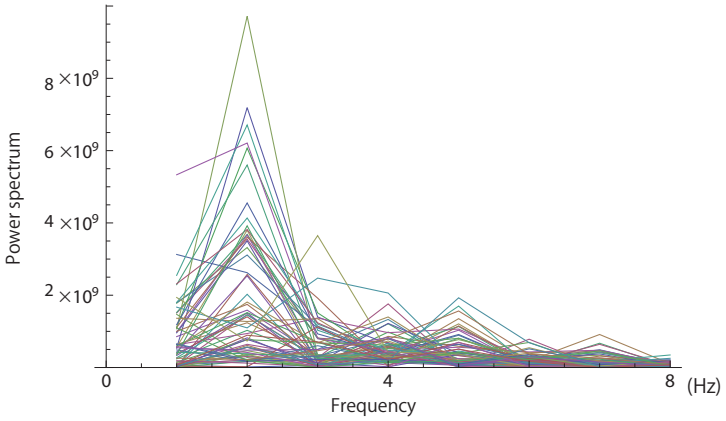


図 4.10: 1 秒間ステップワークにおける屈曲伸展回転角速度のパワースペクトル

入力値

内外転回転角速度のパワースペクトル分析からの入力値として，8 つの周波数帯の中でパワースペクトルの大きい順に，3 つの周波数帯の値を入力値 x_{1t}，x_{2t}，x_{3t} とした．ここで t は t 秒目の入力値であることを示す．また，1 番大きいパワースペクトルの大きさを m_1，2 番目を m_2，3 番目を m_3 とする．そして，パワースペクトルが一番大きい周波数帯が 2 番目，3 番目のパワースペクトルの値と比較した場合に，どのくらい突出しているかを示す値として入力値を x_{4t}，x_{5t} を式 4.3.2, 式 4.3.2 に示す．

$$x_{4t} = \frac{m_2}{m_1} \tag{4.9}$$

$$x_{5t} = \frac{m_3}{m_1} \tag{4.10}$$

とした．この処理を 1 秒間隔ごとに行い，屈曲伸展角速度のパワースペクトル分析の場合も同様の処理を行う．よって，合計 10 の入力値をニューラルネットワークに与える．

学習設定

学習データは学習 1 と同じデータを用いた．学習時の設定は表 4.4 に示す．フィードフォワード型ニューラルネット構成は図 4.11 に示す. 学習回数は 3000 回の学習を繰り返した．3000 回の学習で平均誤差が 0.012, 最大誤差が 0.332 近辺の値で収束した．図 4.11 に示すフィードフォワード型ニューラルネットワークを設定した理由は，試行錯誤，素子数入れ替え学習させ，学習が終束したときの平均誤差と最大誤差が一番小さくなったためである．検証方法は学習 1 と同じ方法を用いた．

表 4.4: 学習 2 の設定

学習結果	3000
入力素子数	10
中間層	1
中間素子数	3
出力素子数	3
ゲイン	0.5
学習データ数	600

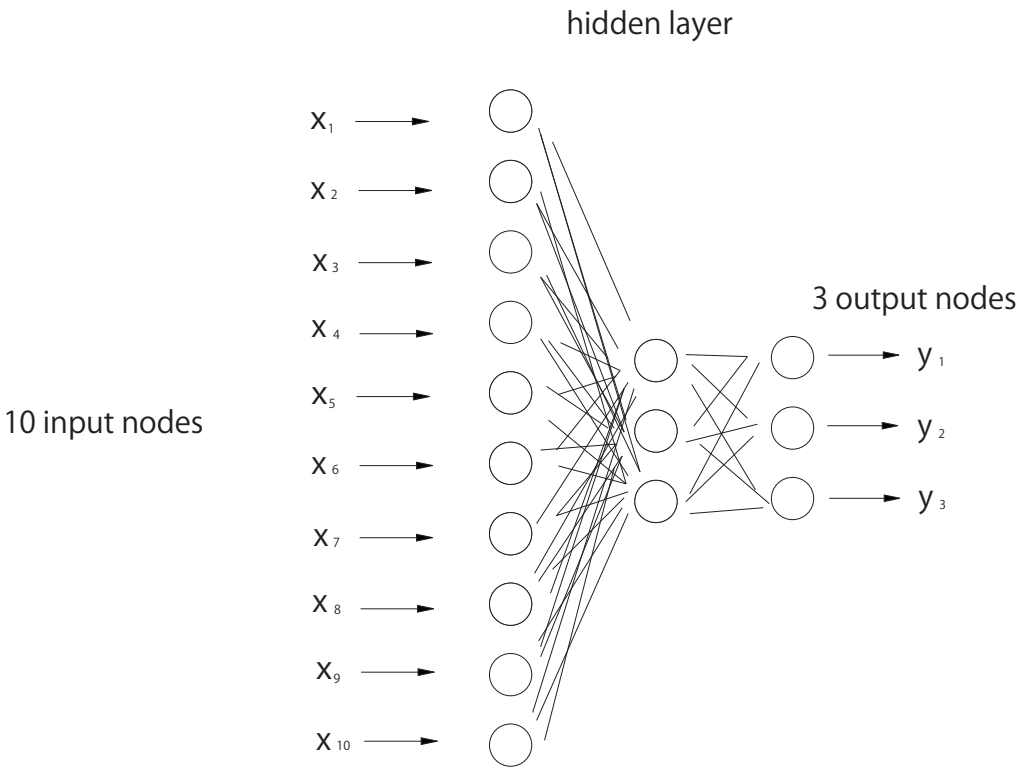


図 4.11: 学習 2 のニューラルネットワーク構成

検証結果

学習 2 の検証結果を表 4.3 に示す.

表 4.5: 学習 2 の検証結果

movements	percentage of correct answers
walk	100%
run	94%
step	90%

4.3.3 考察

学習 1，学習 2 からニューラルネットワークを用いることで，90 %以上の精度で歩き，走り，ステップワークの運動識別パラメーターを獲得することができた．しかし，学習 1 のように入力値に平均値と分散値を与えた学習では，獲得するパラメーターの数が非常に多い．学習 1 で設定したニューラルネットワークの場合，重みとなるパラメーターの数が ω と θ をあわせて 255 個になり，運動識別の計算処理に負荷をかけることになる．また，学習 1 の方法を用いた場合，各モジュールごとにキャリブレーションを行う必要性がある．学習 2 では，入力値にパワースペクトルを用いた分析結果を入力値にあたえることにより，図 4.11 が示すような少ない素子数で，90 %以上の精度の運動識別パラメーターを獲得できた．学習 2 の検証結果ではステップワークの識別精度は学習 1 と比べると低くなっているが，歩行が 100%と非常に高い精度で識別できた．また，素子数が非常に少なくなったため，獲得するパラメーターの数も 45 個になり，学習 1 の時よりもパラメーターの数を 5 分の 1 の程度におさえることができ，運動識別の計算処理を軽減できた．さらに，入力値にパワースペクトルを用いるため，各センサモジュールのキャリブレーションが不要になる．そのため，学習 2 の方法を用いることが，歩き，走り，ステップワークの判別には有効な手段になると考えられる.

4.4 まとめ

ジャイロスコープからの信号を離散フーリエ変換しパワースペクトル分析

からニューラルネットワークの入力値を設定することで，90%以上の精度で歩き，走り，ステップワークの運動識別パラメーターを獲得することができた．このパラメータを制御コンピュータとなる mbed に書き込むことで，ジャイロスコープからの 50Hz の信号を自動的に歩き，走り，ステップワークと識別することができるようになった．

第5章　考察

5.1　はじめに

本章では，3つの考察を行う．1つ目は，開発システムの考察である．本研究では，計測器のプロトタイプを製作し6つの計測実験と2つの機械学習を行った．それらの結果をもとに，開発システムの考察を行う．2つ目はオフザボールの分析についてである．本研究で開発するシステムは主に，オフザボール状況下の分析を対象としている．そこで，開発するシステムがサッカーのオフザボール分析においてどのようなものになるのか考察する．3つ目がオンザボール状況下の分析についてである．本研究で開発するシステムでは，自動的にボールの位置や，選手がボールを保持している状態なのかを分析できるものではない．そのため，オンザボール分析について考察を行う．

.

5.2　開発システム

現在までに，計測器のプロトタイプ製作, 11人の同時計測，GPSを用いた移動軌跡と速度の推定，地磁気センサを用いた体の向きの推定，ジャイロスコープを用いた動作判別をおこなってきた．オフザボールの選手の分析は，選手が『いつ』,『どこで』,『どの向きで』,『なにを』しているのかを分析することである.『いつ』というのは，ボールの位置や味方選手および敵選手の状況のある時刻を示す．そのため，センサを用いてサッカー選手の『いつ』を分析するためには敵味方合わせて22人の選手の運動計測が時刻同期される必要がある．そこで，Zigbeeを用いて各選手の運動計測を同期をさせた．Zigbee PAN 1つで11人の選手を同時計測できる．このPANを2つ構築することにより，敵味方22人の同時計測が可能になる.『どこで』の分析を可能にするのがGPSによる位置計測である．しかし，sup500fモジュールを用いた場合，衛星位置

誤差などの問題から，サッカー選手の位置分析を十分な精度で行うことはできなかった．特に長い時間連続的に計測した場合，位置計測の誤差が大きくなる可能性がある．このため，より高性能で高価な GPS モジュールを用いることでその位置精度を改善でする必要があると考えられる．例えば，DGPS を用いた位置補正が行える Cresent OEM board を用いることにより，位置誤差を 1m 以下に抑えることができる．この Cresent OEM board を陸上競技で利用した場合，その位置精度を 50cm 以内に抑えることができることも報告されている [20]．しかし，低コストな開発を目指し，限られた研究費の中で行った本研究では，Cresent OEM board のような高価の GPS モジュールを用いることができなかった．また，Cresent OEM board のような高精度なモジュールになればなるほど，モジュールの重さとサイズが大きくなる問題がある．ただし，近年の GPS の精度の向上と小型化およびコスト低下を考えると，近い将来安価で高性能な小型 GPS モジュールを使えるようになると考えられる．また，GPS の精度は，GPS の改善やモジュールの改善以外に DGPS や SBAS を用いた補正を行うことで年々その精度は改善される．さらに，2010 年 9 月には日本の準天頂衛星として『みちびき』が打ち上げられるなどの，今後の GPS 精度向上に期待が持てる.『どの向きで』の分析を行うために，地磁気センサ HMC5843 を用いた．HMC5843 を用いた方位計測では，8 つの方位まで推定することができた．8 つの方位が推定できればより詳細なオフザボールの運動分析が可能になる．ただし，サッカー場付近に地磁気の乱れを生み出す電波塔がある場合などの外部障害を考慮する必要がある．ジャイロスコープを用いた運動識別では,『なにを』の分析を可能にする．大腿部の内転外転，屈曲伸展の角速度情報の平均分散または離散フーリエ変換からのパワースペクトル分析による入力値を，フィードフォワード型ニューラルネットワークに与える機械学習を行うことで，歩き，走り，ステップワークのオフザボールの動きを 90%以上の高い精度で識別できた．ただし，ニューラルネットワークの機械学習で用いたデータ数がまだ少ないため，どんなサッカー選手にも，この運動識別が行えるかは不明である．このため，今後より多くの学習データを用いてどんなサッカー選手にも運動識別が行えるようにすることが課題となる．

5.3　オフザボールの分析

サッカーでは，一人の選手がボールを保持する時間は多い選手で 2 分から 3 分である．つまり，オンザボールの状況下よりオフザボールの状況下の方が圧倒的に多いということである．オフザボールの状況下でどのような反応をしているのか，ボールの位置によってどのように動きを変えているか，いつ動きだしているかという，オフザボールの『いつ』『どこで』『どの向きで』『どのような』を分析することは，オンザボールの選手のプレーの失敗成功の原因追求にもなる．本研究で開発したシステムでは，このオフザボールの選手全員の動き『いつ』『どこで』『どの向きで』『どのような』といった動きをリアルタイムで分析することが可能になる．指導者がリアルタイムでこの分析を行うことにより，試合や練習後直後に指導者が選手たちに分析結果を踏まえた的確な指導ができるようになる．位置，速度の推定精度は未だ十分ではないが，この部分が改善されれば，本研究で開発したシステムは，サッカーを指導する現場で非常に有効なツールになるはずである．

5.4　オンザボールの分析

ある選手がオンザボールの状態なのかオフザボールの状態なのかをセンサだけを用いて識別することは非常に困難であった．本研究でも，加速度センサやジャイロスコープを装着しその識別に取り組んだが最適な方法を見つけることが出来なかった．オンザボールの状況は大きく 2 つに分けることが出来る．それは，ドリブルとキックである．ドリブルは数秒間に連続してボールに触れるため，センサを用いてドリブルの状況を運動識別できる可能性はある．しかし，サッカーの熟練度が増すごとに，ドリブルとオフザボールの歩行，走行，ステップワークの運動との違いが小さくなっていく傾向があった．そのため，レベルの高い選手であればあるほど，ドリブルと他の運動との識別が困難になっていくと考えられる．キックの場合，その種類は，パス，シュートに分けることができるが，その動作は一瞬である．そのため，用いるセンサのサンプリングレートを非常に高くする必要がある．また，センサを用いてキック動作を識別できたとしても，実際にその選手がボールを蹴ったかどうかはボールにセンサを付けない限り難しい．なぜなら，サッカー選手はキックフェイントと

いう，蹴る振りをして実際には蹴らないということを頻繁に行うからである．そのため，本研究ではビデオカメラを用いてオンザボールの選手の状況を記録することにした．この場合，ビデオカメラを固定せずにボールを追従する撮影となる．このため，ボール位置がどこにあるのかは自動的に推定できない．しかし，選手全員の位置情報とカメラが同期されることによりそのボール位置は十分推定できると考えられる．近年ではコンピュータヴィジョンの技術を用いてオンザボールの選手がどのような動作をしているのか自動的に判定する研究も行われている [8]．しかし，現状のコンピュータビジョンの方法を用いる場合，ドリブル，パス，シュートまでの判定が限界である．オンザボールの状況下を現場の指導者が分析する場合，右脚を使っているのか，左脚を使っていのか，足の内側でボールに触れているのか，それとも外側なのか，裏側なのかを分析する詳細な着眼点をもっている．つまり，オンザボールの分析では非常に詳細な分析が求められている．そのため，実際の映像を記録しその映像を指導者が観察することがオンザボールの現状の最善な分析になると考えられる．

5.5　まとめ

本章で開発したシステムを用いることにより，サッカー選手のオフザボールの動き，つまり『いつ』『どこで』『どの向きで』『どのような』動きをしているのかを詳細に分析することができる．ただし，GPS 精度の問題からその位置推定には問題が残っている．また，本研究では実際の現場で開発したシステムを運用するところまで至らなかった．そのため，次章の結言で今後の展望を述べると共に，本研究で開発したシステムを実際に運用した場合に考えられる問題をあげる．そして，最後に本研究の総括を述べる．

第6章　結言

6.1　今後の展望

本研究では，筆者の修士課程期間に行った研究であることから，実際の現場で開発したシステムを運用するところまでは至らなかった．そのため，今後このシステムをあるチームで運用し，適切な改良をしていくことが今後の課題となる．現在段階で考えられる課題は2つある．それは，計測機の耐久性と計測データをどのように可視化するかということである．サッカーはコンタクトスポーツであるため，このようなセンサが実際のプレーの障害になることは十分考えられると同時に，センサ自体が破損してしまう可能性もある．そのため，実際の現場の試験運用を通じて，装着位置やセンサを覆うカバーなどを改善していく必要性がある．計測データの可視化方法は図6.1に示すようなwebアプリケーションを考えているが，このアプリケーションのデザインなどは現場の指導者や選手の意見を参考に改善していくことが最善の策であると考えている．

6.2　本研究の総括

本研究では，ジャイロスコープ，GPS，地磁気センサ，Zigbeeを用いて，サッカー選手分析システムの開発に取り組んだ．このような小型センサを用いることで, どこのサッカー場でも低コストで簡易的にサッカー選手の分析が可能になる．開発したプロトタイプではGPSを用いることで，選手の位置，速度を推定し，地磁気センサをもちいることで体の8方向の向きを推定できた．また，ジャイロスコープを選手の右大腿部に装着することで歩き，走り，ステップワークの運動を1秒間隔で90%以上の精度で識別できた．しかし，限られた研究費の中で開発を行ったため，GPSによる位置計測を行った場合，その精度が低下してしまう問題がある．そのため，より精度の高いGPSモジュールを

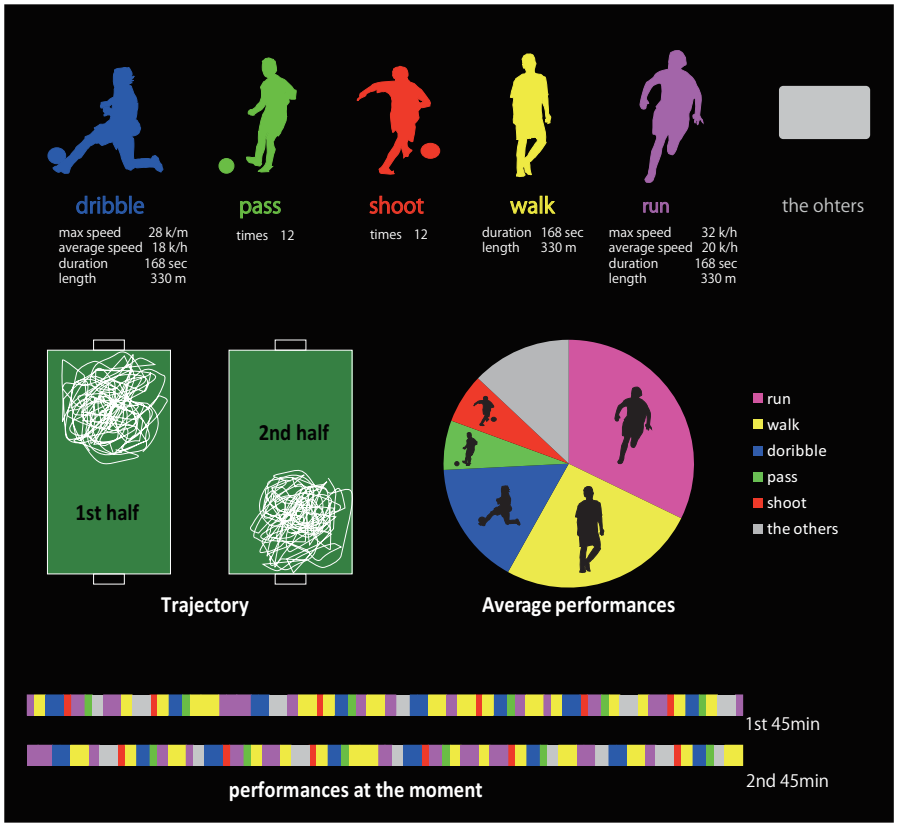


図 6.1: 想定される web アプリケーションの例

用いることでこの問題を解決する必要がある．この位置精度が改善されれば，オフザボールの『いつ』『どこで』『どの向きで』『どのような』動きの分析をより詳細に行うことが可能となり，サッカー指導の現場で非常に有効なツールになる．また，本研究では Zigbee を用いたため，選手全員のオフザボールの動きをリアルタイムに分析することができる．リアルタイムで分析することにより，指導者は試合や練習後すぐ選手たちに分析結果を踏まえた的確な指導ができるようになる．

参考文献

[1] TRACAB，http://www.tracab.com/default.asp (2011 年 1 月 10 日確認).

[2] 加藤久，サッカーの戦術とコンピューター分析の現状と課題.1999

[3] 大橋二郎，サッカーのゲーム分析用リアルタイムデータ入力プログラムの試作．日本体育協会スポーツ科学研究報告 No.10 ボールゲームの分析法に関する研究－第 1 報－，pp.17—23，1987.

[4] 大橋二郎，サッカーのリアルタイムパス分析システムの実用化．日本体育協会スポーツ科学研究報告，No.8 ボールゲームの分析法に関する研究－第 2 報－，pp.4—9，1988.

[5] 大橋二郎，選手の動きの分析，J.J.Sport Sci2—10，pp. 785—793，1983.

[6] 戸刈晴彦，サッカーのゲーム分析－リアルタイム処理法による－．体育の科学 36—9，pp. 699—703，1990.

[7] 大串暫朗，ビデオを利用したゲーム分析，J.J.SportSci2—10，pp. 794—800，1983

[8] 中川靖士，サッカー映像の自動ゲーム分析方法の提案と評価, UNISYS TECHNOLOGY REVIEW 第 76 号, FEB. 2003.

[9] 中川靖士, 羽田久一, 今井正和, 砂原秀樹，サッカー映像の自動ゲーム分析, 情報処理学会研究報告マルチメディア通信と分散処理, 2002—DPS—106：193—198, 2002.

[10] Akito TAKEYA，Development of Formation Camera System for Stadium, 日本光学会 36 8, pp.466-474, 2007.

[11] 市川 晃, 竹家章仁，全方位視覚システムの最適設計, 計測自動制御学会論文集 35, pp. 1243—1252, 1999.

[12] FIFA.com，http://www.fifa.com/ (2011 年 1 月 10 日確認).

[13] Oshima Y, Kawaguchi K, Tanaka S, Ohkawara K, Hikihara Y, Ishikawa-Takata K, Tabata I，Classifying household and locomotive activities using a triaxial accelerometer., Gait & Posture31, pp. 370-374, 2010

[14] Masahiro Tada, Ren Ohmura, Masaya Okada, Futoshi Naya, Haruo Noma, Tomoji Toriyama, Kiyoshi Kogure, A Method for Measuring and Analyzing Driving Behavior Using 3D-Accelerometers

[15] 藤岩秀樹，サッカーゲームにおける失点シーンの特徴, 宇部工業高等専門学校研究報告 52 号, pp83-88, 2006.

[16] 藤岩秀樹:，サッカーゲームにおける得点シーンの分析，運動とスポーツの科学 10，pp.75—80，2004.

[17] Ricardo M. L. Barros, Milton S. Misuta 1, Rafael P. Menezes 1, Pascual J. Figueroa, Felipe A. Moura, Sergio A Cunha, Ricardo Anido, Neucimar J. Leite，Analysis of the distances covered by first division Brazilian soccer players obtained with an automatic tracking method, Journal of Sports Science and Medicine 6, pp.233-242, 2007.

[18] Misuta, M.S., Menezes, R.P., Figueroa, P.J., Cunha, S.A.，Representation and analysis of soccer players' trajectories., 20th Congress of the International Society of Biomechanics, pp.415, 2005.

[19] 久保信明, 喬耘, 安田 明生，GPS 単独測位高精度化の実現性について, 日本航海学会論文集 (114), pp.259-266, 2006.

[20] 長嶋拓哉, 高橋隆行，GPS を用いた陸上競技者リアルタイム計測システムの開発, 計測自動制御学会東北支部第 253 回研究集会資料番号 253-7, 2009.

[21] 坂井丈泰，GPS 技術入門（東京電機大学出版局), 2003.

[22] 太田憲, 仰木裕嗣, 木村広, 廣津信義，スポーツデータ（共立出版）, 2005

[23] 武藤佳恭，ニューラルコンピューティング（コロナ社）, 1996

[24] 熊沢逸夫，学習とニューラルネットワーク（森北出版）, 1998

[25] James A. Freeman, SIMULATING NEURAL NETWORKS (Addison-Wesley Publishing Company), 1994.

対外発表

渡辺　俊，仰木　裕嗣，マルチセンサを用いたサッカー選手分析システムの開発，シンポジウム：スポーツ・アンド・ヒューマン・ダイナミクス 2010, pp.157-162, 2010.

謝辞

主査 仰木先生 学部2年生の時から4年間，時には厳しく，それでも厳しくのご指導頂けたことを本当に感謝しています．あれは愛のムチだったと信じています．私にとって仰木研はスポーツの視点を大きくかえてくれ,『探求』のおもしろさを教えてもらえてた場所でした．今後の仰木研のご発展を期待しています．4年間本当にありがとうございました．

副査 三次先生 ご多忙の中，研究手法，論文執筆だけでなくモノ作り側の視点からの貴重なご意見本当にありがとうございました

副査 永野先生 ご多忙の中，研究手法だけでなく，サッカー競技者からの視点の貴重なご意見本当にありがとうございました．

金田さん 真剣になると，時には仰木先生より鋭い視点で斬り込んでくる仰木研の斬り込み隊長．参考になる意見をたくさん頂けました．今後の研究者としてのご活躍を期待しています．本当にありがとうございました．

ばしさん 常に年したの後輩に気を配り，面倒をみてくれる良き先輩．最初で最後の牛丼は一生忘れません．今後の研究者としてのご活躍を期待しています．本当にありがとうございました．

ケンさん 2年生から生意気な後輩として，サッカー仲間として，研究仲間として，ケンさんとはいろんな関係が築けました．本当にありがとうございました．

レノさん 『おい！レノ!!』そんな暴言を吐いてすいませんでした．最後の修論添削を付きっきりでやってくれるレノさんの優しさ．そして，ご褒美ロールケーキまで買ってきてくれるその優しさ．一生忘れません．本当にありがとうございました．

たけしげ 最後の印刷のときだって徹夜で手伝ってくれるいつも優しいお兄ちゃん．タメなんだけどね．本当にありました[1]．

ななこ ななこがきてからロフトがなんだか騒がしい．でも，とても楽しい空間になった気がする．ざんまいサラダおごらなきゃ〜〜．本当にありがとうございました．

父母 SFC で充実した生活がおくれたのは，間違いなく親父とお母のおかげです．今まで僕の未来のために惜しみない投資をしてくれたことに感謝しています．本当にありがとうございました．

[1]北京家常菜宏金流感謝表現

付 録A

mbedに用いたソースコード

```
#include "mbed.h"
#include <string>
#include <math.h>
#include <string>
#include <stdio.h>
#include "direction.h"
#include "motion.h"
#include "position.h"

#define mID 1//モジュール ID
#define pi 3.1415926535//πの設定

Serial pc(USBTX, USBRX); // USB シリアルの設定
I2C i2c_2(p28, p27);   //HMC5843 との通信に用いるための I2Cpin 設定
Serial gps(p9, p10); //sup500f との通信に用いるシリアル pin 設定
Serial xbee(p13, p14); //Xbee との通信に用いるシリアル pin 設定
DigitalOut rst1(p11); //Xbee のリセットピン設定
AnalogIn x(p19);//LPY5150AL の X 軸データ読み込みのアナログ pin 設定
AnalogIn z(p20);//LPY5150AL の Z 軸データ読み込みのアナログ pin 設定

//HMC5843 の設定に用いるアドレス定義
const int address = 0x3C;
const int CTRL_REGA = 0x00;
const int CTRL_REGB = 0x01;
const int MODE_REG = 0x02;
const int XOUT_H = 0x03;
const int XOUT_L = 0x04;
const int YOUT_H = 0x05;
const int YOUT_L = 0x06;
const int ZOUT_H = 0x07;
const int ZOUT_L = 0x08;
const int STATUS = 0x09;

//sup500F の設定に用いるためのデータ配列の定義
unsigned char gps_def[9];
unsigned char gps_band[11];
unsigned char gps_nmea[16];
unsigned char gps_output[10];
unsigned char gps_sr[10];
unsigned char gps_waas[10];
unsigned char gps_nm[10];
```

```
string name = "";
string utc = "";
string status = "";
string lat = "";
string ns = "";
string lon = "";
string ew = "";
string speed = "";

int16_t valx;
int16_t valy;
int16_t valz;

int high, low, sta;
int counter = 0;
int times = 0;
int split = 0;
int dir = 0;
int mot = 0;
int mx1, mx2, mx3, mz1, mz2, mz3;
int pmx1, pmx2, pmx3, pmz1, pmz2, pmz3;

float poX = 0.0;
float poY = 0.0;
float vel = 0.0;
float rx, rz;
float pmxn2, pmxn3, pmzn2, pmzn3;

double ReFx[8],ImFx[8],ReFz[8],ImFz[8], spex[8], spez[8];
double mxn1, mxn2, mxn3, mzn1, mzn2, mzn3;

Ticker flipper;//割り込みの定義

//サンプリングレート 50Hz でデータを獲得するための割り込み
void flip(){

    times++;
    rx = x.read()*10;
    rz = z.read()*10;
    for(int i = 0; i < 8; i++){
        ReFx[i]+=rx*cos(2*pi*times*(i+1)/50);
        ImFx[i]+=-rx*sin(2*pi*times*(i+1)/50);
        ReFz[i]+=rz*cos(2*pi*times*(i+1)/50);
        ImFz[i]+=-rz*sin(2*pi*times*(i+1)/50);
    }

//LPY5150AL から 50 回分（1 秒間）のデータを獲得
    if(times == 50){
        for(int i = 0; i < 8; i++){
            spex[i]= ReFx[i]*ReFx[i] + ImFx[i]*ImFx[i];
            spez[i]= ReFz[i]*ReFz[i] + ImFz[i]*ImFz[i];
        }

        for(int i = 0; i < 8; i++){
            if(mxn3 <= spex[i] && mxn2 <= spex[i] && mxn1 <= spex[i] ){
```

```
            mxn3 = mxn2;
            mx3 = mx2;
            mxn2 = mxn1;
            mx2 = mx1;
            mxn1 = spex[i];
            mx1 = i+1;
        }
        else if(mxn3 <= spex[i] && mxn2 <= spex[i] && mxn1 > spex[i] ){
            mxn3 = mxn2;
            mx3 = mx2;
            mxn2 = spex[i];
            mx2 = i+1;
        }
        else if(mxn3 <= spex[i] && mxn2 > spex[i] && mxn1 > spex[i] ){
            mxn3 = spex[i];
            mx3 = i+1;
        }

        if(mzn3 <= spez[i] && mzn2 <= spez[i] && mzn1 <= spez[i] ){
            mzn3 = mzn2;
            mz3 = mz2;
            mzn2 = mzn1;
            mz2 = mz1;
            mzn1 = spez[i];
            mz1 = i+1;
        }
        else if(mzn3 <= spez[i] && mzn2 <= spez[i] && mzn1 > spez[i] ){
            mzn3 = mzn2;
            mz3 = mz2;
            mzn2 = spez[i];
            mz2 = i+1;
        }
        else if(mzn3 <= spez[i] && mzn2 > spez[i] && mzn1 > spez[i] ){
            mzn3 = spez[i];
            mz3 = i+1;
        }
    }

    pmx1 = mx1;//ニューラルネットの入力値 x1
    pmx2 = mx2;//ニューラルネットの入力値 x2
    pmx3 = mx3;//ニューラルネットの入力値 x3
    pmz1 = mz1;//ニューラルネットの入力値 x4
    pmz2 = mz2;//ニューラルネットの入力値 x5
    pmz3 = mz3;//ニューラルネットの入力値 x6
    pmxn2 = mxn2/mxn1;//ニューラルネットの入力値 x7
    pmxn3 = mxn3/mxn1;//ニューラルネットの入力値 x8
    pmzn2 = mzn2/mzn1;//ニューラルネットの入力値 x9
    pmzn3 = mzn3/mzn1;//ニューラルネットの入力値 x10

//運動識別
mot = motion(pmx1, pmx2, pmx3, pmz1, pmz2, pmz3, pmxn2, pmxn3, pmzn2, pmz3);

    //HMC5843 からデータを獲得する
    char data = XOUT_H;
    i2c_2.write(address, &data, 1);
```

```
        i2c_2.read(address, &data, 1);
        high = data;

        data = XOUT_L;
        i2c_2.write(address, &data, 1);
        i2c_2.read(address, &data, 1);
        low = data;

        valx = (high<<8) | low;

        data = YOUT_H;
        i2c_2.write(address, &data, 1);
        i2c_2.read(address, &data, 1);
        high = data;

        data = YOUT_L;
        i2c_2.write(address, &data, 1);
        i2c_2.read(address, &data, 1);
        low = data;

        valy = (high<<8) | low;

        data = ZOUT_H;
        i2c_2.write(address, &data, 1);
        i2c_2.read(address, &data, 1);
        high = data;

        data = ZOUT_L;
        i2c_2.write(address, &data, 1);
        i2c_2.read(address, &data, 1);
        low = data;

        valz = (high<<8) | low;

dir = direction(valx, valy, valz);//方位推定

        for(int i = 0; i < 8; i++){
            ReFx[i]=ImFx[i]=ReFz[i]=ImFz[i]=0.0;
        };

        mxn1 = mxn2 = mxn3 = mzn1 = mzn2 = mzn3 =0.0;//初期化
        mx1 = mx2 = mx3 = mz1 = mz2 = mz3 =0;//初期化
        times = 0;

        counter++;
    }

}


int main() {
///HMC5843 設定
    char data1[2] = {MODE_REG, 0};//HMC5843 スタート
    i2c_2.write(address, data1, 2);
```

```
    char data2[2] = {CTRL_REGA, 0x08};//サンプリングレートの設定
    i2c_2.write(address, data2, 2);

    char data3[2] = {CTRL_REGB, 0x20};// レンジ設定
    i2c_2.write(address, data3, 2);

    //Xbee のリセット
    rst1 = 0;
    wait_ms(1);
    rst1 = 1;
    wait_ms(1);

    xbee.printf("Type any charactor\n");//計測 PC に命令

    while(xbee.readable() == 0);//計測 PC から，なにか文字を受信したら
    xbee.getc();

//GPS　スタート
    gps_def[0]=0xA0;
    gps_def[1]=0xA1;
    gps_def[2]=0x00;
    gps_def[3]=0x02;
    gps_def[4]=0x04;
    gps_def[5]=0x01;
    gps_def[6]=0x05;
    gps_def[7]=0x0D;
    gps_def[8]=0x0A;

//シリアル通信の Baudrate 設定 9600
    gps_band[0]=0xA0;
    gps_band[1]=0xA1;
    gps_band[2]=0x00;
    gps_band[3]=0x04;
    gps_band[4]=0x05;
    gps_band[5]=0x00;
    gps_band[6]=0x01;
    gps_band[7]=0x01;
    gps_band[8]=0x05;
    gps_band[9]=0x0D;
    gps_band[10]=0x0A;

//データ型設定　RMC
    gps_nmea[0]=0xA0;
    gps_nmea[1]=0xA1;
    gps_nmea[2]=0x00;
    gps_nmea[3]=0x09;
    gps_nmea[4]=0x08;
    gps_nmea[5]=0x00;
    gps_nmea[6]=0x00;
    gps_nmea[7]=0x00;
    gps_nmea[8]=0x00;
    gps_nmea[9]=0x01;
    gps_nmea[10]=0x00;
    gps_nmea[11]=0x00;
    gps_nmea[12]=0x01;
```

```
    gps_nmea[13]=0x08;
    gps_nmea[14]=0x0D;
    gps_nmea[15]=0x0A;

//NMEA メッセージの設定
    gps_output[0]=0xA0;
    gps_output[1]=0xA1;
    gps_output[2]=0x00;
    gps_output[3]=0x03;
    gps_output[4]=0x09;
    gps_output[5]=0x01;
    gps_output[6]=0x01;
    gps_output[7]=0x09;
    gps_output[8]=0x0D;
    gps_output[9]=0x0A;

//サンプリングレートの設定 1Hz
    gps_sr[0]=0xA0;
    gps_sr[1]=0xA1;
    gps_sr[2]=0x00;
    gps_sr[3]=0x03;
    gps_sr[4]=0x0E;
    gps_sr[5]=0x01;
    gps_sr[6]=0x01;
    gps_sr[7]=0x0E;
    gps_sr[8]=0x0D;
    gps_sr[9]=0x0A;

//WAAS の設定
    gps_waas[0]=0xA0;
    gps_waas[1]=0xA1;
    gps_waas[2]=0x00;
    gps_waas[3]=0x03;
    gps_waas[4]=0x37;
    gps_waas[5]=0x01;
    gps_waas[6]=0x01;
    gps_waas[7]=0x37;
    gps_waas[8]=0x0D;
    gps_waas[9]=0x0A;

    //移動モードの設定
    gps_nm[0]=0xA0;
    gps_nm[1]=0xA1;
    gps_nm[2]=0x00;
    gps_nm[3]=0x03;
    gps_nm[4]=0x3C;
    gps_nm[5]=0x01;
    gps_nm[6]=0x01;
    gps_nm[7]=0x3C;
    gps_nm[8]=0x0D;
    gps_nm[9]=0x0A;

    //sup500f の設定を送信
    for(int i = 0; i<9; i++){
        gps.putc(gps_def[i]);
```

```
    }
    xbee.printf("gps.def OK\n");
    wait(1);

    for(int i = 0; i<11; i++){
        gps.putc(gps_band[i]);
    }
    xbee.printf("gps.band OK\n");
    wait(1);

    for(int i = 0; i<16; i++){
        gps.putc(gps_nmea[i]);
    }
    xbee.printf("gps.nmea OK\n");
    wait(1);

    for(int i = 0; i<10; i++){
        gps.putc(gps_output[i]);
    }
    xbee.printf("gps.output OK\n");
    wait(1);

    for(int i = 0; i<10; i++){
        gps.putc(gps_sr[i]);
    }
    xbee.printf("gps.sr OK\n");
    wait(1);

    for(int i = 0; i<10; i++){
        gps.putc(gps_waas[i]);
    }
    xbee.printf("gps.waas OK\n");
    wait(1);

    for(int i = 0; i<10; i++){
        gps.putc(gps_nm[i]);
    }
    xbee.printf("gps.nm OK\n");
    wait(1);

flipper.attach(&flip, 0.02);//割り込みを 0.02 秒間隔で行う

    while(1) {
        char r = 0;

//sup500f からデータを獲得（改行がくるまで 1 文字ずつ獲得する）
        while(r != 0x0A && gps.readable() == 1){
            r = gps.getc();

            if(r == 0x2c){
                split++;
            }

//獲得した RMC データを分割
            if(r != 0x2c){
```

```
                if(split == 0)name += r;
                else if (split == 1)utc += r;
                else if (split == 2)status += r;
                else if (split == 3)lat += r;//緯度
                else if (split == 4)ns += r;
                else if (split == 5)lon += r;//経度
                else if (split == 6)ew += r;
                else if (split == 7)speed += r;//速度
            }
}

if(r == 0x0A){
poX = positionX(lat, lon);
poY = positionY(lat, lon);
vel = velocity(speed);

//データを計測 PC に送信
            xbee.printf( "%d,%d,%d,%d,%f,%f,%f\n",mID, counter, dir, mot, poX, poY, vel);

            name = "";
            utc = "";
            status = "";
            lat = "";
            ns = "";
            lon = "";
            ew = "";
            speed = "";
            split = 0;
            }
      }
}
```

付 録B

計測PCに用いたソースコード

```
import processing.video.*;
import java.io.File;
import processing.serial.*;
import java.io.*;
import processing.net.*;
import de.bezier.data.sql.*;

StringBuffer sb = new StringBuffer();
String stringData;

PrintWriter output;
Capture myCapture;
Serial port;
Client client;
PostgreSQL pgsql;

int m = 0;
int k = 0;
int index = 1;
int count = 1;

boolean start = false;

void setup() {
File dir = new File("保存ディレクトリの指定");
if (!dir.exists()) {
dir.mkdir();
}



//画面サイズ設定
size(400, 400);
output = createWriter("csv ファイルの保存ファイル名.csv");

port=new Serial(this,"シリアルポート指定",9600);
port.clear();
port.bufferUntil(0x0A); //改行まで読み込む


//キャプチャする映像の設定
myCapture = new Capture(this, width, height, 1);
```

```
  //データベースの設定
String user     = "ユーザ名";
String pass     = "パスワード";
String database = "データベース名";
pgsql = new PostgreSQL( this, "IP アドレス:ポート番号", database, user, pass );

frameRate(1);
}

void draw() {
//映像を画面に配置
image(myCapture, 0, 0);
if(start){
//記録画像を（jpg で保存);
save("p" + index + ".jpg");

index++;
}
}



void keyPressed(){
//画像の保存開始
if(key==’s’){
start=true;
}
//e キーで記録終了
if(key==’e’){
start=false;
}
//c キーでカメラセッティング
if(key==’c’){
myCapture.settings();
}

if(key == ’t’){
port.write(’a’);//mbed と通信開始
}

//r キー 1 回で保存終了，2 回ですべて終了
else if (key == ’r’){
m++;
if(m == 1){
output.flush();
output.close();
println("Save completely");
}

if(m == 2)
exit();
}
}
```

```
//映像の読み込み
void captureEvent(Capture myCapture) {
myCapture.read();
}

//シリアルデータ読み込み
void serialEvent(Serial p) {
if(m == 0){
stringData = port.readStringUntil(0x0A);
stringData = trim(stringData);// 改行を取り除く

String list[] = split(stringData, ',');

if(start && list.length() == 7){

//mbed からのデータを保存
output.print(index + ","  + stringData);

//データベースに送信
if (pgsql.connect())
{
if(count == 1){
String tableName = "TABLE"+ nf(year(), 4) + nf(month(), 2)
 + nf(day(), 2) + nf(hour(), 2) + nf(minute(), 2);

pgsql.execute("CREATE TABLE " + tableName +
 " ( index INTEGER, ID INTEGER, counts INTEGER,
 Direction INTEGER, WM INTEGER, pX FLOAT,
 pY FLOAT, Speed FLOAT);");
}

pgsql.execute("INSERT INTO " + tableName +
 " (index, ID, counts, Direction, WM, pX, pY, Speed)
VALUES (" + "'" + index + "'" + "," + "'" + list[0] + "'"
 + "," + "'" + list[1] + "'" + "," + "'" + list[2] + "'" + ","
+ "'" + list[3] + "'" + "," + "'" + list[4] + "'" + "," + "'"
+ list[5] + "'" + "," + "'" + list[6] + "'" + ");");
count++;
}


}
else{
print("not save");
}

}
port.clear();

}
```

付 録C

フィードフォワード型ニューラルネットワークの学習に用いたソースコード

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

#define NUM_LEARN 学習の繰り返しの最大回数
#define NUM_SAMPLE 訓練データの sample 数
#define NUM_INPUT 入力ノードの数
#define NUM_HIDDEN 中間層の素子数
#define NUM_OUTPUT 出力素子数
#define EPSILON 学習時の重み修正の程度を決める
#define THRESHOLD_ERROR_MAX 学習誤差の閾値でこの値以下なら学習を終了する
#define THRESHOLD_ERROR_COUNT 学習誤差がこの値以上ならカウントする
#define GAIN 学習誤差の閾値でこの値以下なら学習を終了する

//訓練データを配列に格納する
float tx[NUM_SAMPLE][NUM_INPUT], ty[NUM_SAMPLE][NUM_OUTPUT];

//x は入力　 h は中間層，y は出力，x，y が＋ 1 されるのは，閾値 θ を決めるため.
float x[NUM_INPUT + 1], h1[NUM_HIDDEN + 1], h2[NUM_HIDDEN + 1], y[NUM_OUTPUT];

//重み． 1 つ余分なのは閾値を決めるため.
float w1[NUM_INPUT + 1][NUM_HIDDEN], w2[NUM_HIDDEN + 1][NUM_HIDDEN], w3[NUM_HIDDEN + 1][NUM_OUTPUT];

//中間層，出力素子の逆伝搬量.
float h_back1[NUM_HIDDEN + 1], h_back2[NUM_HIDDEN + 1], y_back[NUM_OUTPUT];


int main (int argc, const char * argv[]) {

char *fname = "読み込む学習 CSV ファイルの指定";
char *Wfname = "学習を終了した時の重みを記録するファイルの指定";
int ilearn, isample, i, j;
float in[4], out[4];
float net_input, error, max_error, ave_error, epsilon, seed;
int count_error;
int inet_input;
FILE *fp;

epsilon = (float)EPSILON;
isample = 0;
```

```
//訓練データの読み込み
fp = fopen( fname, "r" );
if( fp == NULL ){
printf( "%s ファイルが開けません\n", fname );
return -1;
}

while( (fscanf( fp, "%f,%f,%f,%f,%f,%f,%f,%f",
 &in[0], &in[1], &in[2], &in[3], &out[0], &out[1], &out[2], &out[3] ) ) != EOF ){
for(j = 0; j < NUM_INPUT; j++){
tx[isample][j] = (float)(log10((double)in[j]));
}
for(j = 0; j < NUM_OUTPUT; j++){
ty[isample][j] = out[j];
}
printf( "NO:%d  入力:%f %f %f %f  出力:%f %f %f %f",
isample+1, tx[isample][0], tx[isample][1], tx[isample][2], tx[isample][3],
ty[isample][0], ty[isample][1], ty[isample][2], ty[isample][3]);
printf( "\n");
isample++;

}
printf("%d\n", isample);
fclose( fp );

//重みの初期設定
seed =(float)1;
for(i = 0; i < NUM_INPUT + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
seed = seed * (float)-1;
w1[i][j] = seed;
}
}
for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
seed = seed * (float)-1;
w2[i][j] = seed;

}

}
for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_OUTPUT; j++){
seed = seed * (float)-1;
w3[i][j] = seed;

}

}


//学習ループ
for(ilearn = 0; ilearn < NUM_LEARN; ilearn++){
```

```
max_error = 0;
ave_error = 0;
count_error = 0;

//訓練データに関するループ
for(isample = 0; isample < NUM_SAMPLE; isample++){
for(i = 0; i < NUM_INPUT; i++){
x[i] = tx[isample][i];
}
x[NUM_INPUT] = (float)1.0;//閾値用の入力として最後の配列を 1.0 とする

//中間層素子の計算
for(j = 0; j < NUM_HIDDEN; j++){
net_input = 0;
for(i = 0; i < NUM_INPUT + 1; i++){
net_input = net_input + w1[i][j] * x[i];
}

h1[j] = (float)((double)1.0 / ((double)1.0 + exp((double)net_input * (double)GAIN * (double)-1.0)));
}
h1[NUM_HIDDEN] = (float)1.0;//閾値用の入力として最後の配列を 1.0 とする

for(j = 0; j < NUM_HIDDEN; j++){
net_input = 0;
for(i = 0; i < NUM_HIDDEN + 1; i++){
net_input = net_input + w2[i][j] * h1[i];
}

h2[j] = (float)((double)1.0 / ((double)1.0 + exp((double)net_input * (double)GAIN * (double)-1.0)));
}
h2[NUM_HIDDEN] = (float)1.0;//閾値用の入力として最後の配列を 1.0 とする

//出力素子の計算
for(j = 0; j < NUM_OUTPUT; j++){
net_input = 0;
for(i = 0; i < NUM_HIDDEN + 1; i++){
net_input = net_input + w3[i][j] * h2[i];
}

y[j] = (float)((double)1.0 / ((double)1.0 + exp((double)net_input * (double)GAIN * (double)-1.0)));
}

//誤差の評価
error = 0;
for(j = 0; j < NUM_OUTPUT; j++){
error = error + pow((ty[isample][j] - y[j]), 2);
}
error = error / (float)NUM_OUTPUT;
if(error > max_error){
max_error = error;
}

ave_error = ave_error + error;

if(error > (float)THRESHOLD_ERROR_COUNT){
```

```
count_error++;
}

//バックプロパゲーション
for(j = 0; j < NUM_OUTPUT; j++){
y_back[j] = 2 * (y[j] - ty[isample][j]) * ((float)1.0 - y[j]) * y[j];
}
for(i = 0; i < NUM_HIDDEN; i++){
net_input = 0;
for(j = 0; j < NUM_OUTPUT; j++){
net_input = net_input + w3[i][j] * y_back[j];
}
h_back2[i] = net_input * ((float)1.0 - h2[i]) * h2[i];
}
for(i = 0; i < NUM_HIDDEN; i++){
net_input = 0;
for(j = 0; j < NUM_HIDDEN; j++){
net_input = net_input + w2[i][j] * h_back2[j];
}
h_back1[i] = net_input * ((float)1.0 - h1[i]) * h1[i];
}


//重みの修正
if(max_error >= (float)THRESHOLD_ERROR_MAX || isample+1 != NUM_SAMPLE){
for(i = 0; i < NUM_INPUT + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
w1[i][j] = w1[i][j] - epsilon * x[i] * h_back1[j];
}
}

for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
w2[i][j] = w2[i][j] - epsilon * h1[i] * h_back2[j];
}
}

for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_OUTPUT; j++){
w3[i][j] = w3[i][j] - epsilon * h2[i] * y_back[j];
}
}
}




}//訓練ループ終了
printf("NO = %d, max = %f ave = %f count = %d\n", ilearn+1, max_error,
ave_error/(float)NUM_SAMPLE, count_error);

if(max_error < (float)THRESHOLD_ERROR_MAX){
break;
}
```

```
}//学習ループ終了
//最終的な重みを表示

fp = fopen( Wfname, "w" );
if( fp == NULL ){
printf( "%s ファイルが開けません\n", Wfname );
return -1;
}

printf("学習回数 = %d, 最高誤差  = %f \n", ilearn + 1, max_error);
fprintf( fp, "%s,%d\n", "学習回数", ilearn + 1);
fprintf( fp, "%s,%f\n", "最大誤差", max_error);
fprintf( fp, "%s,%f\n", "平均誤差", ave_error/(float)NUM_SAMPLE);
fprintf( fp, "%s,%d\n", "指定した閾値を超えた誤差の数", count_error);


for(i = 0; i < NUM_INPUT + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
printf("w1[%d][%d] = %f \n", i, j, w1[i][j]);
fprintf( fp, "%d,%d,%d,%f\n", 1,i, j, w1[i][j]);
}
}

for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_HIDDEN; j++){
printf("w2[%d][%d] = %f \n", i, j, w2[i][j]);
fprintf( fp, "%d,%d,%d,%f\n", 2,i, j, w2[i][j]);
}
}

for(i = 0; i < NUM_HIDDEN + 1; i++){
for(j = 0; j < NUM_OUTPUT; j++){
printf("w3[%d][%d] = %f \n", i, j, w3[i][j]);
fprintf( fp, "%d,%d,%d,%f\n", 3,i, j, w3[i][j]);
}
}


fclose( fp );

printf( "%s ファイル書き込みが終わりました\n", fname );
    return 0;
}
```