

道路の区画ID方式に基づくヒューマンプローブにおける 低消費電力な測位手法の提案

細川 茂樹^{1,a)} 藤波 香織^{2,b)}

概要：携帯端末やセンサデバイスの小型化・高性能化により，個人が気温や騒音等の環境情報を収集し，他人と共有するヒューマンプローブの研究が盛んに行なわれている．しかし，継続的な測位が必要なヒューマンプローブにおいては，センサデバイスの省電力性の確保が課題となる．本稿では，道路の区画ID方式に基づくヒューマンプローブに適用可能な低消費電力測位基盤システムを提案する．既存手法の測位精度と省電力性の向上をめざし，2つの測位アルゴリズム GPS_PE (1区間に対しGPS測位1回)，GPS_PE_CC (GPS_PEにて1区間に対し2回測位される問題を解消)を考案した．シミュレーション評価の結果，関連研究の手法に比べて測位誤差を平均12.8 [%]改善し，消費電力を100~200 [mW]削減した．

キーワード：ヒューマンプローブ，スマートフォン，省電力化，GPS，デッドレコニング

1. はじめに

近年，スマートフォンやセンサデバイスの小型化・高性能化により，個人が気温や騒音等の環境情報を収集し，他人と共有するヒューマンプローブの研究が盛んにおこなわれている [1]．ヒューマンプローブにおけるセンシング方法はユーザの関わり方により日和見センシング (Opportunistic Sensing) と参加型センシング (Participatory Sensing) の2種類に分けられる．日和見センシングは外環境の情報を自動で収集するセンシング方法であり，気温，湿度 [11][25]，道路の騒音 [2]，路面状況 (凹凸 [4]，凍結 [5]，降雨 [26]) などの自動収集に用いられる．日和見センシングは自動でデータを収集するため，ユーザがデータ収集を意識する必要は無い．一方，参加型センシング (Participatory Sensing) はユーザが手動でデータを収集するセンシング方法であり，センサでは取得できない情報 (PM2.5を排出するディーゼル車の分布 [3] など) を収集する際に用いられる．これらのヒューマンプローブにて収集したデータは，インターネットを通じてヒューマンプローブの他の参加者と共有する．

しかし，日和見センシングはセンサデータを収集する時

表 1 スマートフォン (Galaxy Nexus) の消費電流

項目	消費電流 [mA]
GPS 測位時	55.000
3G (通信時)	150.000
Wi-Fi (通信時)	150.000
Bluetooth (通信時)	150.000
加速度センサ	0.139
地磁気センサ	4.000
角速度センサ	10.239

表 2 外付けセンサの消費電流の例

センサ種類	消費電流 [mA]
音センサ (ADMP401[13])	0.250
温度センサ (SHT-71[14])	0.500

空間の範囲が広く，多くのデータを収集する必要があるため，処理が継続的に実施されることが多い．ゆえに，スマートフォンの消費電力が大きくなる問題がある．表 1 に，スマートフォンの消費電力の一例として，power_profile.xml から取得した Samsung Galaxy Nexus SD-04 の消費電流を示す．内蔵されたセンサだけでなく，スマートフォンの拡張ポートにセンサを外付けして，計測を行うこともある [25][26]．表 2 に外付けセンサデバイス用のセンサの消費電流の例を示す．これらの表から，GPS や 3G などを用いる無線通信処理は消費電力が大きいことが分かる．

一方，近年では道路を交差点で区切った道路区画を空間的な単位として扱う研究が多い [2][4][11]．これは，ユーザである歩行者は道路上を歩き，情報も道路に沿って収集されるためと考えられる．また，OpenStreetMap[23] など道

¹ 東京農工大学大学院 工学府 情報工学専攻
Department of Computer and Information Sciences, Tokyo
University of Agriculture and Technology

² 東京農工大学大学院 工学研究院 先端情報科学部門
Department of Computer and Information Sciences, Tokyo
University of Agriculture and Technology

a) hososhige0526@gmail.com

b) fujinami@cc.tuat.ac.jp

表 3 分布図の表現形式とナビゲーション(経路探索)における課題

表現方法	データ収集面積	経路探索
等高線	広い	道路との関連付けが必要
メッシュ	広い	道路との関連付けが必要
道路区画	狭い	可能

路 DB の一般公開も関連している。これらの背景から、一般財団法人日本デジタル道路地図協会では、道路上の参照点と参照点からの距離を用いて位置情報を取り扱う方式「道路の区間 ID 方式」を提案している [15]。また、倉沢らは特定の時空間において収集したセンサデータの精度を一定以上保証するためのセンサデータ収集手法を提案している [12]。これらの例のように、道路区画に付加した ID (道路 ID) を用いて位置を表現する仕組みや応用手法が整備されつつある。また、交差点をノード、道路区画をリンクとして扱い、災害情報のリアルタイム共有や交通ナビゲーションシステムへ活用する試みも行われている [2][4][11]。さらに、計測データの可視化方法としては、分布図(等高線、メッシュ [5]、道路区画 [2][4][11]) が多く用いられる。分布図の表現形式の違いを表 3 に示す。データ収集面積が狭いほど収集データ数が少なくなり、省電力化に貢献できるが、列挙した形式の中では道路区画が最も狭い。また、センサデータを用いたナビゲーションでは、歩行者は道路上のみを歩くことがほとんどであるため、道路上のデータだけ収集すればよいことになる。したがって、本研究では歩行者からのデータ収集との親和性、地図データ利用環境の整備、省電力化の観点から、道路区画を単位とした分布図を対象とする。

以上の背景を踏まえ、本研究では道路の区画 ID (道路 ID) 方式に基づくヒューマンプローブにおける、センサノードの省電力稼働の実現を目的とする。なお、前提条件として、システムの対象者を歩行者と定め、歩行者はスマートフォンを用いて屋外を移動しているものとする。

本論文の構成について述べる。まず、本節で本研究の背景と目的を述べた後、第 2 節にて端末の省電力化戦略に関する関連研究および本研究の立ち位置を述べる。次に、第 3 節で測位基盤システムの設計について詳説し、続く第 4 節で実装方法を説明する。これらの節をもとに実装したシステムについて、第 5 節ではシミュレーション評価と結果について述べ、第 6 節では実環境における実験の方法および結果と考察を詳述する。最後に、第 7 節にて本研究の結論を述べる。

2. 関連研究

2.1 GPS 測位に関する省電力化戦略

表 1 で示したように、GPS は消費電力が多く、基地局への頻繁なアクセスはスマートフォンの消費電力増加につながる。ゆえに、GPS の利用機会をできる限り減らすための研究が行われてきた [6][7][8]。Baun, et al. はユーザの静止

状態を加速度で検知し、測位誤差が閾値以下に保たれるスリープ時間を設定する戦略を提案している [7]。この戦略は Zhuang, et al.[6] が示した GPS スリープ戦略(SS) と類似している。Ben, et al. の研究でも、Zhuang らの SS, SR 戦略と同様の戦略を提案している [8]。しかし、これらの測位手法を用いて GPS をスリープすると、道路 ID の変化が考慮されずスリープが実行されるため、道路 ID の変動を見逃す可能性がある。この場合、GPS スリープ中は正確な位置情報が得られない。

2.2 デッドレコニング(自律型測位)

デッドレコニングはデバイス内蔵のセンサを用いる測位方法である。3 軸加速度センサで歩数計測などを行って移動距離を算出し、地磁気センサにより移動方向を特定して現在位置を推定する [9][21]。デッドレコニングはセンサの消費電力が少ないが、測位誤差が蓄積していくため時間経過に伴い誤差が大きくなる。本研究では、GPS との複合測位によりデッドレコニングの誤差を補う。

2.3 複合型測位の省電力化戦略

Hopfner, et al.[10] は GPS と加速度センサを用いて最近傍の交差点までの距離を推定し、交差点に到達するまで GPS をスリープさせ、携帯端末の消費電力を削減する複合型の測位手法を提案している。彼らはユーザの歩行速度を 1) 一定、2) 一定かつ静止状態を考慮、3) GPS で得た過去 2 箇所の座標から推定、4) 加速度センサで得た歩幅から算出、の 4 パターンで推定する方法を提案している。しかし、彼らの測位手法は交差点付近で GPS 測位回数が増加する傾向があるため、交差点が多い場所(都市や住宅街など)における計測では、携帯端末の消費電力が大きくなる問題が発生する。本研究では、彼らの手法をベースに、交差点付近においても GPS 測位回数が増加しない測位手法を提案する。また、彼らの研究では道路形状や歩行速度の違いを考慮した実験が行われていなかった。本研究では、道路形状だけでなく歩行者の移動速度の違いにも着目し、複数の道路状態パターンに応じた測位手法を提案する。

3. 測位基盤システムの設計と実装

3.1 道路 ID 設定方式

1 節で述べたように、ヒューマンプローブの研究ではセンサデータを道路単位の分布図で表現することが多い [2][4][11]。これらの研究を分析した結果、道路 ID の設定方法に次の共通点があることが分かった。

- (1) 道路を分岐点で分割
- (2) 分割した道路を「道路区画」と定義(図 1(a))
- (3) 区画が長い場合は一定距離で分割(分割区画)(図 1(b))

本研究では、これらの関連研究に適用するため、上記の道路 ID 設定方式に基づき測位基盤システムを設計する。

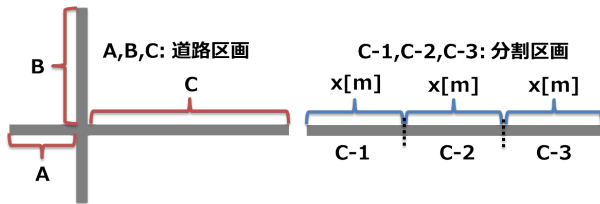


図 1 道路区画

3.2 システム構成

Hopfner, et al. の測位手法は現時点における道路座標を用いた最新の測位手法である。彼らのアルゴリズムの方針は、道路の交差点でのみ GPS 測位を行い、歩行者が所有する携帯端末の消費電力を抑えることである。Hopfner, et al. の測位アルゴリズムを、本研究の提案方式と関連付けて説明する。彼らの測位アルゴリズムは「GPS 測位中」、「道路 ID 変換中」、「移動距離推定中 (GPS スリープ中)」の 3 つの状態を持つ。「GPS 測位中」では GPS により歩行者の緯度経度を取得した後、「道路 ID 変換中」状態に移移する。「道路 ID 変換中」の時は、緯度経度を道路 ID に変換し、歩行者のいる道路を特定する。その後、「移動距離推定中」状態に移移し、GPS を待機させつつ歩行者の移動距離を推定する。

本研究では、彼らの測位手法を継承して、図 2 に示すような新たな測位基盤システムを提案する。本システムは歩行者が歩行時に保有する携帯端末上で実行され、GPS および加速度センサが搭載されているものとする。

- 測位モジュール
 - 携帯端末の測位機能を利用して歩行者の緯度経度を取得し、道路 ID に変換する。
 - 緯度経度取得モジュール
 - 歩行者の現在位置を緯度経度で取得し、道路 ID 変換モジュールに提供する。
 - 道路 ID 変換モジュール
 - 道路 DB とマップマッチングを利用し、歩行者の位置情報を道路 ID に変換する。
 - 道路 DB
 - 歩行者座標をもとに道路の座標を検索・抽出し、道路 ID 変換モジュールに提供する。
 - 移動距離推定モジュール
 - 次の GPS 測位地点までの距離を 1 秒ごとに推定し、距離が一定以下になるまで GPS をスリープさせる。
 - 測位アルゴリズム状態管理モジュール
 - 測位アルゴリズムの 3 つの状態「GPS 測位中」「道路 ID 変換中」「移動距離推定中」を切り替える。
- 道路 ID 提供モジュール
 - 道路 ID 変換モジュールから道路 ID を取得し、アプリケーションに提供する。

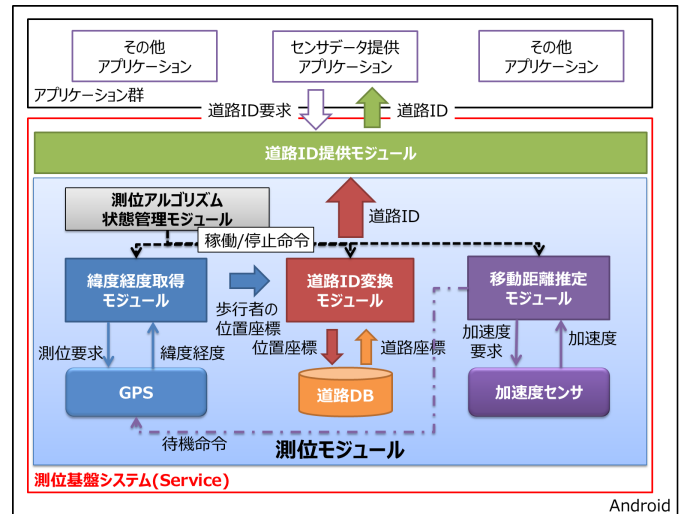


図 2 測位基盤システムの構成

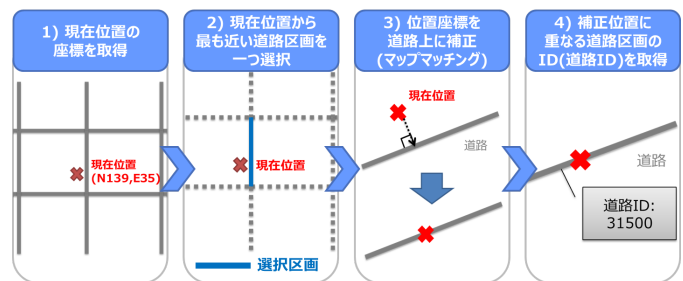


図 3 道路 ID 変換方法

表 4 道路テーブルの例

道路 ID	分割 ID	道路位置情報 (始端と終端の緯度経度)
1	1	(39.00 135.00, 39.01 135.01)
1	2	(39.01 135.01, 39.02 135.02)

3.3 道路 ID 変換モジュール

測位モジュールの構成要素である道路 ID 変換モジュールの定義について説明する。

3.3.1 道路 ID 変換方法

位置座標を道路 ID へ変換する方法を、図 3 を用いて説明する。まず、1) 端末の位置座標を GPS で測位して用意する。次に、2) その位置から最も近い道路区画を DB の道路テーブルから検索、抽出する。その後、3) DB から得られた道路区画の集合を用いてマップマッチング [26] を行い、位置座標を道路上に補正する。最後に、4) マップマッチングで補正された位置座標と重なる道路区画を歩行者のいる道路区画とみなし、その道路 ID を取得する。

3.3.2 道路 DB の設計

道路 DB には表 4 のような道路テーブルが含まれている。道路 ID は道路区画毎に割り振られる ID である。分割 ID は道路区画を複数に分割した際の列挙番号である。道路の位置情報は DB に標準搭載されている関数により、緯度経度形式に変換して取得できる。

3.3.3 マップマッチング

マップマッチングは地図 (道路) データを用いて現在位置のずれを補正する手法である。Brakatsoulas, et al.[16]

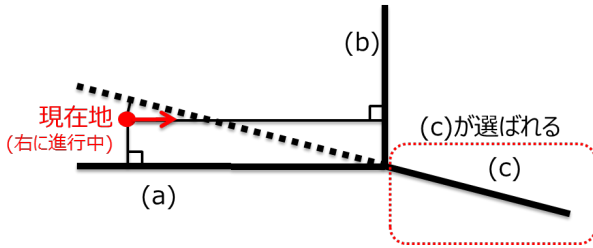


図 4 Brakatsoulas, et al. のマップマッチングが失敗する例

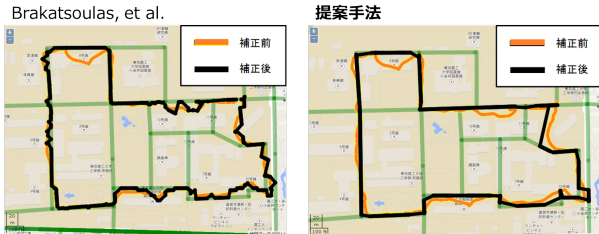


図 5 評価関数修正前 (左) と修正後 (右) のマップマッチング結果

のマップマッチング手法では、現在位置と移動方向、道路区画の位置情報を用いる。現在位置 p_i と道路区画 e_j との距離を $d(p_i, e_j)$ 、現在の進行方向と e_j がなす角を a_{ij} とする。これらのパラメータを用いて式 (1), (2) により評価値 S を算出する。評価値 S は (1) と (2) の和で得られる式 (3) の値である。評価値が最大となる道路区画 e_j をマッチング先として選定し、現在位置から e_j へ降ろした垂線の足へ点を移動させる。

$$S_d(p_i, e_j) = 10 - 0.17d(p_i, e_j)^{1.4} \quad (1)$$

$$S_a(p_i, e_j) = 10\cos^4(a_{ij}) \quad (2)$$

$$S(p_i, e_j) = S_d(p_i, e_j) + S_a(p_i, e_j) \quad (3)$$

我々の提案手法では GPS スリープ中に移動距離を推定するが、Brakatsoulas, et al. のように方位センサ (地磁気センサ) を用いると、誤差により計測値が不安定になり、移動距離が正しく推定できないことが分かった。そのため、提案手法では移動方向を考慮しない。この場合、図 4 の状況において、歩行者は道路 (c) 上にあるものとみなし、位置が誤って補正される。そこで、距離パラメータ $d(p_i, e_j)$ を、垂線距離ではなく道路区画の重心までの直線距離に変更する。変更後のマップマッチング結果を図 5 に示す。

3.4 測位モジュール

本研究では、Hopfner, et al. の測位アルゴリズムを改良し、新たな測位アルゴリズムを提案する。

3.4.1 Hopfner, et al. の測位アルゴリズムの問題点

Hopfner, et al.[10] の測位アルゴリズムは、屋外の歩行者の位置情報を継続して把握するために提案されたアルゴリズムである。Hopfner, et al. のアルゴリズムのうち、dynamic は直近の 2 つの GPS 測位点から移動速度を求め、monitoring は加速度センサにより歩数と歩幅を推定し、移動速度および移動距離を算出する。この移動距離が

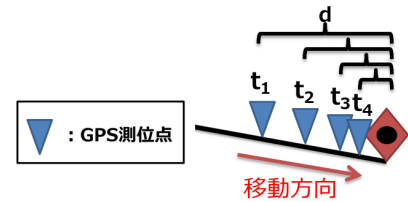


図 6 交差点付近で GPS 測位回数が増大するイメージ図

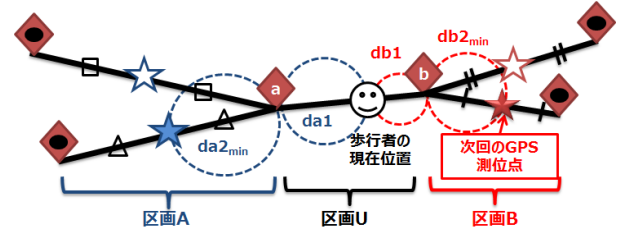


図 7 提案測位アルゴリズム (GPS_PE) の可視化

次の GPS 測位地点までの距離に近づいた場合、状態管理モジュールは「GPS 測位中」状態に切り替える。しかし、Hopfner, et al. の測位アルゴリズムは、現在位置が分岐点に近づくとき図 6 に示すように移動距離 d が徐々に短くなり、GPS 測位回数が増大するという問題がある。

3.4.2 GPS_PE アルゴリズム

このため、上述の Hopfner, et al. の測位アルゴリズムの問題点を解決する、新たな測位アルゴリズム GPS_PE (GPS と Pedometer を組み合わせた測位の意) を提案する。GPS_PE は測位精度と省電力性が高く、歩行者の移動速度を最も正しく反映する monitoring アルゴリズムを継承したアルゴリズムである。GPS_PE は次の GPS 測位位置として、歩行者から最短距離にある隣接区画の中心に定める。これを図 7 を用いて詳説する。歩行者は区画 U に存在するとしたとき、道路区画の両端 a, b までの距離をそれぞれ $da1, db1$ とする。また、端 a, b に接続する道路区画の中心 (星マーク) と端 a, b までの距離 $da2, db2$ のうち、最小距離をそれぞれ $da2_{min}, db2_{min}$ と定義する。 $(da1 + da2_{min})$ と $(db1 + db2_{min})$ を比較し、小さい方の距離を持つ中心を次回の GPS 測位点として定める。小さい距離を採用する理由は、2 つ隣の区画で測位が発生する状況を防ぐためである。この変更により、次の GPS 測位位置までの距離が長くなり、GPS 測位回数が減る。GPS_PE のアルゴリズムを Algorithm1, Algorithm2 に示す。

3.4.3 GPS_PE の改良

GPS_PE の測位アルゴリズムは、隣接区画の中に短い区画が混入すると、図 8 (左) に示すように 1 つの区画につき 2 回以上 GPS 測位が発生する。この問題に対し、図 8 (右) のように、GPS スリープ中の移動距離を移動開始時の区画中心から移動終了時の区画中心までの距離に変更する。この変更により図 8 の距離 $d1$ が長くなり、GPS スリープ中の移動距離が長くなるため、同一区画内の重複測位が起きず、電力を保持できる。この変更を適用した GPS_PE を GPS_PE.CC (Center to Center) として提案する。

Algorithm 1 次回の GPS 測位点を決定する

```

Input: (x, y) ∈ Lname // 現在位置
a, b // 区画 U の端点 a, b
aseg // 点 a を端に持つ道路区画の集合 (歩行者の区画を除く)
bseg // 点 b を端に持つ道路区画の集合 (歩行者の区画を除く)
Output: PE' (Lname) // GPS スリープ終点
def gps.next.active.point((x, y), a, b, aseg, bseg):
    dmin = ∞
    da1 = distance((x,y), a) // 現在位置から端 a までの距離
    db1 = distance((x,y), b) // 現在位置から端 b までの距離
    (la_min, i_min) = shortest_seg_id.len(aseg) // A の長さ ID
    (lb_min, j_min) = shortest_seg_id.len(bseg) // B の長さ ID
    da2_min = la_min / 2 // 端 a から区画 A 中心までの距離
    db2_min = lb_min / 2 // 端 b から最短 B 中心までの距離
    d1 = da1 + da2_min // 現在位置から区画 A 中心までの距離
    d2 = db1 + db2_min // 現在位置から区画 B 中心までの距離
    minsegid = null // 現在地から最短距離にある区画の ID
    if d1 >= d2 then
        d = d1
        minsegid = i_min
    else
        d = d2
        minsegid = j_min
    end if
    PE' (Lname) = center(seg(minsegid)) // 最近傍区画の中心
    return (PE' (Lname))

```

Algorithm 2 点 P に接する最短隣接区画 ID および長さを返す

```

Input: pseg // 点 p を端に持つ道路区画の集合
Output: minid, l_min // 最短の区画 ID およびその長さ
def shortest_seg_id.len (pseg):
    l_min = ∞ // 最短区画の長さ
    minid = -1 // 最短区画の ID
    for each pseg(i) ∈ pseg do
        li = length(pseg(i))
        if l_min > li then
            l_min = li
            minid = i // 最短区画 ID を更新
        end if
    end for
    return (minid, l_min) // 最短区画 ID とその区画長を返す

```

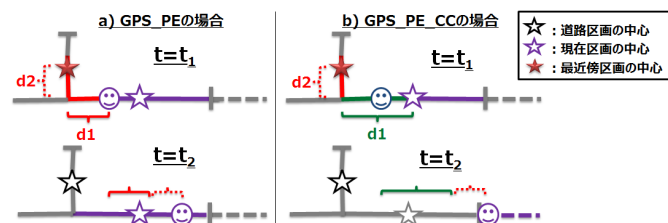


図 8 提案手法の差異: a) GPS_PE, b) GPS_PE_CC

表 5 スマートフォン (Galaxy Nexus) の仕様

名称	GALAXY NEXUS SC-04D
OS (カーネルバージョン)	Android 4.2.2 (3.0.31-g9f818de)
バッテリー容量	1,750 mAh
CPU	Texas Instruments OMAP4460 1.2GHz (DualCore)

3.5 測位システムの実装

システムを搭載する端末として Samsung Electronics Co., Ltd. の GALAXY NEXUS SC-04D[17] を採用し、Android 上で動作する Service として実装した。端末の仕様を表 5 に、ソフトウェアの開発および実行環境を表 6 に示す。なお、道路 DB は Android に標準搭載の SQLite に道路データの格納、検索、抽出といった地図データ処理のためのプラグイン Spatialite[18] を追加したものを利用する。

表 6 ソフトウェア開発・実行環境

開発ライブラリ	Java Development Kit 7 Android SDK 4.2.2 (API 15) Spatialite 4.1.1
DB	SQLite (Spatialite プラグイン含む)

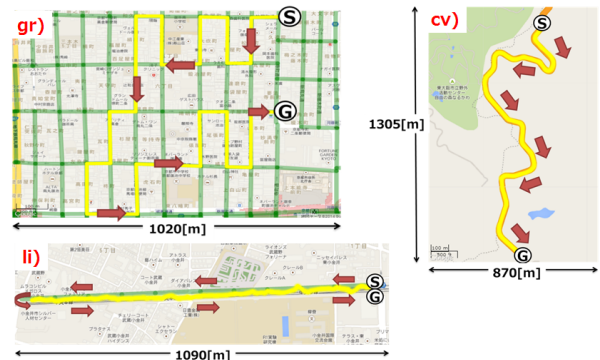


図 9 経路 (gr. 基盤目状, cv. カーブ, li. 直線)

4. シミュレーション評価

4.1 目的および評価方法

本研究で提案する測位基盤システムが要件を満たすことを検証する。具体的には、歩行者が歩行シナリオに基づいて歩行した場合の、本提案手法の測位精度および消費電力を、Hopfner, et al.[10] の測位手法と比較してシミュレータにより評価する。測位誤差は正解区画の中心から GPS 測位位置までの直線距離を表し、短いほど精度が高い。マップマッチング成功率は区画 ID と正解区画 ID との一致度を表し、高いほうが望ましい。歩行シナリオは道路形状別に 3 種類用意した。1 つ目のシナリオは、京都の基盤目状の経路を歩くシナリオである。経路図を図 9 (gr) に示す。黄線は経路を表し、緑線は道路区画を表す。また、カーブの道路を歩く場面を想定し、図 9 (cv) に示すような奈良県鳴川峠 (なるかわとうげ) を歩くシナリオを用意した。道路は 100 [m] ごとの曲線区画に区切った。最後のシナリオは、東京農工大学工学部キャンパス北の線路沿い直線道路を一往復するシナリオである。経路図は図 9 (li) に示す通りであり、経路は 100 [m] ごとに区切った。

4.2 シミュレータ概要

シミュレータは筆者が独自に開発したシミュレータを用いる。このシミュレータは歩行者の速度やその変動幅、移動経路 (位置情報の配列)、GPS 測位の平均誤差および誤差変動幅等のパラメータを入力すると、一秒毎の歩行者の道路 ID や位置座標、時刻、測位アルゴリズムの各状態の処理時間を算出し、テキストファイルに出力する。これらの出力情報を用いて測位精度と消費電力の評価を行う。消費電力については、GPS 測位、マップマッチング、移動距離推定の 3 つの処理の所要時間と消費電流 [mA]、バッテリーの動作電圧 [V] から平均消費電力 [W] を算出する。

表 7 各処理の消費電流のパラメータ

処理名	消費電流 [mA]
GPS 測位 (捕捉時および捕捉後)	260 (380)
GPS 測位 (非補足時)	0 (120)
マップマッチング	195 (315)
移動距離推定	10 (130)
Service	120

表 8 測位時の内部処理の所要時間

処理名	所要時間 [秒/回]
GPS 測位	5.0
マップマッチング	1.0

表 9 測位精度に関するパラメータ

パラメータ名	値 [単位]
GPS の平均測位誤差	15.0 [m]
歩幅	0.77 [m]
移動速度誤差	± 20.0 [%]

4.3 パラメータ設定

表 7 は測位アルゴリズムの各処理の消費電流を表している。この消費電流は Galaxy Nexus にデジタルマルチメータを接続して計測した平均値である。本システムは Android の Service 稼働時の消費電流を加えた値 (小かつこの値) を用いる。表 7 の最後に示す Service 稼働時の消費電力は、Activity と Service を稼働している時の消費電流から、Activity のみ稼働している時の消費電流を引いた値である。表 8 は、Java の時刻取得関数を用いて計測して得た各処理の所要時間である。バッテリー電圧は Android SDK 標準搭載の `BatteryManager` クラスの関数を用いて取得した値の平均値 4.0[V] を用いた。表 9 に示すとおり、GPS の平均測位誤差は肥田ら [20] に従って半径 15 [m] に設定した。歩幅は実験者の身長から 1 [m] 引いた 0.77 [m] に設定した [9]。移動速度の誤差は、実験前にあらかじめ収集した歩行データの速度誤差を設定した。ユーザの歩行速度は 0.665 [m/s], 1.330 [m/s], 1.995 [m/s] の 3 通りに設定した。標準速度を 1.330 [m/s] に設定した理由は、「不動産の表示に関する公正競争規約 (表示規約)」[19] において、徒歩 1 分が 80 [m] に相当すると定義されているためである。

4.4 評価結果

4.4.1 測位誤差

図 10 に移動速度および道路形状別の測位誤差とマップマッチングの成功率を示す。図 10 (左) より、測位誤差は道路形状に関係なく GPS_PE が最小であった。提案手法の平均測位誤差は、Hopfner, et al. の手法の平均測位誤差より平均 4.6 [m] (12.8%) 短縮された。グラフのエラーバーは標準偏差を表すが、GPS_PE, GPS_PE_CC はいずれもばらつきが少なくなった。また、提案手法のマップマッチング成功率は Hopfner, et al. の手法の平均成功率に比べて 4.6 [%] 改善した。図 10 (右) により、道路区画長にばら

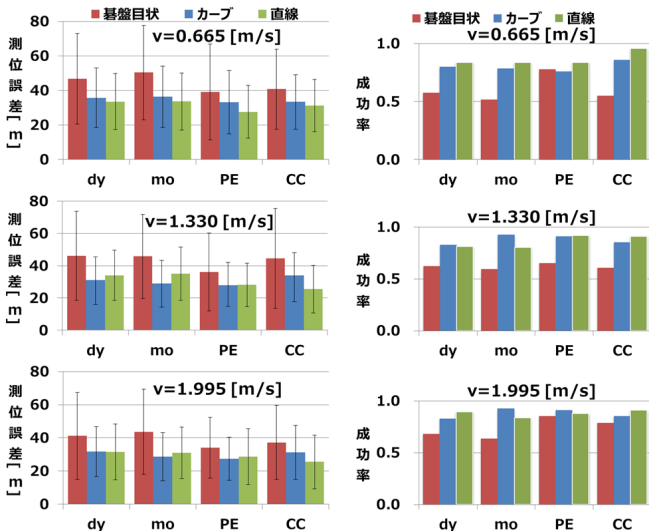


図 10 測位誤差 (左) とマップマッチング成功率 (右)

表 10 バッテリー駆動予想時間 [h] (基盤目状/カーブ/直線)

移動速度 上 : 0.665 [m/s], 中 : 1.330 [m/s], 下 : 1.995 [m/s]

dynamic	monitoring	GPS_PE	GPS_PE_CC
7.4/8.5/7.2	10.3/10.2/10.3	12.2/11.9/12.3	12.3/12.3/12.4
9.1/7.4/8.3	8.9/8.7/8.4	11.2/10.8/11.3	11.4/11.4/11.5
8.3/7.6/7.5	8.1/7.8/7.8	10.4/9.8/10.5	10.9/10.4/10.7

つきがある基盤目状の成功率が低いことから、道路区画長の分散が小さいほど成功すると考えられる。同様に、道路区画が密集していた基盤目状の成功率が低いことから、道路区画が密集しない場所ほど成功すると推測できる。アルゴリズム別では、いずれの状況においても GPS_PE が最高となった。

4.4.2 消費電力

図 11 に移動速度および道路形状別の端末の消費電力および消費電力量を示す。GPS_PE, GPS_PE_CC の消費電力は、dynamic, monitoring の平均消費電力と比べて 100~200 [mW] (平均 13.1 [%]) 減少した。これは 1,750 [mAh] のバッテリーにおいて駆動時間を約 2 時間延長したこととなる (表 11)。また、図 11 より、道路形状の差異による消費電力の差は確認されなかった。アルゴリズム別では、GPS_PE_CC が最も消費電力が少なく、次いで GPS_PE が少なかった。この傾向は移動速度および道路形状の種類に影響を受けていない。一方、電気消費量の内訳を表す図 11 (右) から、提案手法により GPS 測位時間を削減できたが、加速度センサを用いた移動距離推定処理がボトルネックであることがうかがえる。

5. 実環境実験

5.1 目的と評価方法

本評価では、シミュレーション評価にて定数として扱った移動速度、GPS 測位時間が変動した場合の、提案測位アルゴリズムの測位誤差および消費電力を評価する。測位システムの実装を実装した携帯端末を保持

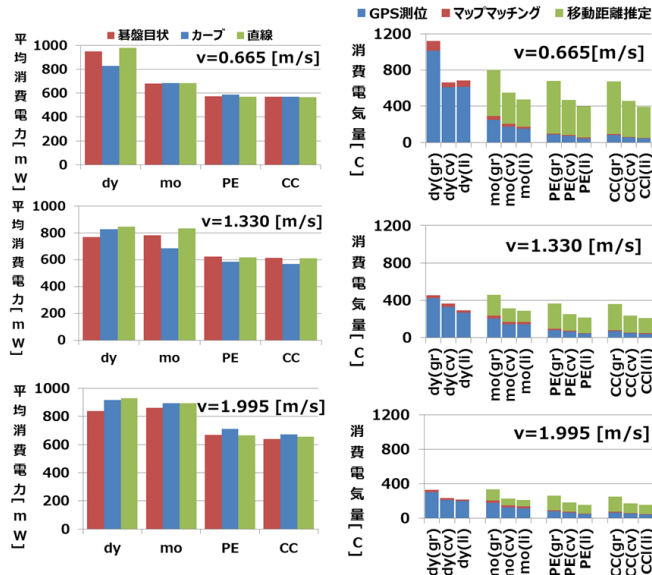


図 11 消費電力(左)と消費電力量(右)
(gr: 基盤目状, cv: カーブ, li: 直線)

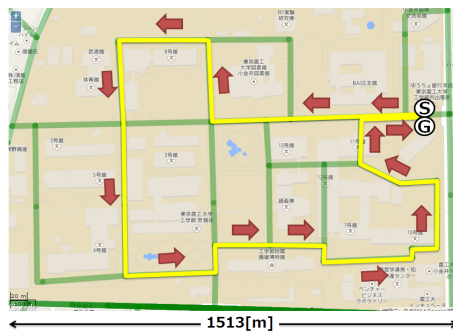


図 12 実環境実験における経路

し、屋外を歩行する。携帯端末は、右利きの歩行者が歩行する場面を想定し、歩行者のズボン右前ポケットに格納した。本実験では、図 12 に示すような東京農工大学工学部キャンパス内の一定経路を歩行するシナリオを用意した。ここで歩行者は黄色い線を赤い矢印に従って歩行するものとする。道路区画数は 22 個であり、道路区画長の平均は 66.7 m、標準偏差は 30.1 m である。歩数については、歩行ステップを認識する閾値をあらかじめズボン前ポケット格納時に合わせて調整した。具体的には、加速度ベクトルノルムが 0.9 G 以上になってから 1 秒以内に 0.9 G 以下に下がった場合に、1 ステップと認識するように設定した。この設定における歩数推定誤差は 4.5% である。GPS 測位の許容誤差は最大 15 m に設定した [20]。

5.2 結果

図 13 は GPS_PE_CC 利用時におけるマップマッチング後の位置を可視化した図である。区画上の青点は道路にマップマッチングされた後の位置を表す。また、赤い点線の円で囲まれた箇所は、GPS 測位が行われず通過した区画であり、22 個中 5 個の区画の測定が失敗したことを表す。また、シミュレーション(机上)と比較した実環境に

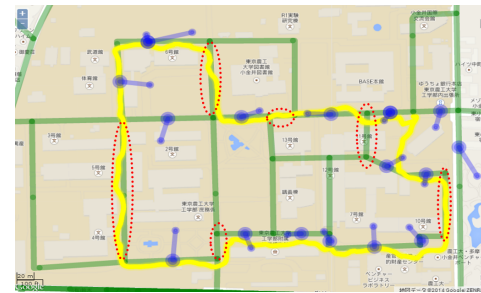


図 13 GPS 測位位置 (GPS_PE_CC)

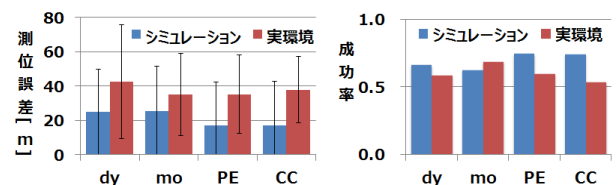


図 14 測位誤差(左)、マップマッチング成功率(右)

表 11 バッテリー駆動予想時間 [h]

環境	dynamic	monitoring	GPS_PE	GPS_PE_CC
机上	6.4	8.0	10.5	10.9
実環境	4.1	4.3	4.7	5.3

おける測位誤差およびマップマッチング成功率を図 14 に示し、バッテリー駆動予想時間を表 11 に示す。測位誤差は実環境の方が大きく、マップマッチングも実環境の方が失敗していることが分かった。バッテリー駆動予想時間は表 11 に示すように、実環境の方が 2~6 時間少なかった。

5.3 考察

5.3.1 マップマッチング飛躍の原因と解決策

マップマッチング飛躍の原因は、GPS を捕捉する処理に時間がかかるためである。本実験における GPS の測位時間は約 7~9 秒(標準偏差 4~15 秒)であり、ばらつきが大きかった。解決策として、道路区画長が GPS 測位中の移動距離より短く、飛躍が発生しやすい状況では GPS を常時捕捉しておき、道路区画長が長い場合は GPS を毎回捕捉し直す方法が考えられる。ただし、2018 年に準天頂衛星システムが整備された場合 [24]、測位時間短縮が実現され、マップマッチング飛躍の発生を軽減できると考える。

5.3.2 測位誤差の原因と解決策

実環境において測位精度に課題があることが分かった。測位誤差の要因として、GPS 測位時間の推定誤差、歩数推定誤差、歩幅推定誤差があげられる。このうち歩幅推定誤差は歩行速度、年齢、性別、身長、地面の勾配など、様々な要因により発生し、誤差が生じやすいため、特に重要である。歩幅推定誤差の向上に向けては、高精度な歩幅推定手法の導入を検討している。歩行速度を考慮した歩幅推定法の例として、重回帰分析を用いて個人の歩行周波数と歩幅の関係式を推定する手法が利用されており [9][22]、低速歩行と高速歩行の歩幅変化を考慮できる。ただし、図 11 から分かるように、本提案手法は加速度センサによる消費

電力の割合が大きいため、消費電力を考慮した手法を導入する必要がある。また、ユーザが自身の歩数や歩幅を手入力で設定する場合、手間が多くなるため、入力処理を自動化する必要がある。

6. おわりに

本論文では、道路の区画 ID 方式に基づくヒューマンプロブに適用できる低消費電力な測位システムを提案した。設計においては省電力性向上と測位精度向上の両立を考慮して、緯度経度取得機能、道路 ID 判定機能、GPS スリープ中の移動距離推定機能の実現方法について検討した。その結果、二つの測位アルゴリズム GPS_PE (1 区間につき GPS 測位 1 回)、GPS_PE_CC (GPS_PE にて 1 区間につき 2 回測位される問題を解消) を考案した。シミュレーション評価により低消費電力化の有効性を示した後、実環境実験を実施した。提案手法は、以下の項目を達成した。

- 関連研究の測位手法の消費電力を 100~200 [mW] 削減
- 携帯端末のバッテリー残量を平均 13.1 [%] 削減
- 道路形状に依存せず、関連研究の既存手法よりも測位誤差を平均 4.6 m 短縮 (12.8 [%] 改善)
- 道路 ID 判定精度を平均 4.6 [%] 改善
- 測位精度重視の時は GPS_PE が適し、消費電力重視の時は GPS_PE_CC が適することを明らかにした

特に最後の項目は、アルゴリズムを動的に切り換える機構の有効性を示唆している。今後は、測位精度の向上や個人に特化した測位パラメータの自動設定、状況に適した測位アルゴリズムへの動的切替機能の追加により、実用性の高い測位システムの実現を目指す。

参考文献

- [1] J. Goldman, et al.: "Participatory Sensing: A Citizen-powered Approach to Illuminating the Patterns That Shape our World", *Foresight and Governance Project*, White Paper, 2009.
- [2] R. K. Rana, et al.: "Ear-phone: an end-to-end participatory urban noise mapping system", In *Proceedings of IPSN 2010*, pp.105-116, 2010.
- [3] T. S. Lena, et al.: "Elemental Carbon and PM2.5 in an Urban Community Heavily Impacted by Diesel Truck Traffic", *Environmental Health Perspectives*, Vol.110, pp.1009-1015, 2012.
- [4] 野村智洋, 他; "スマートフォンを用いた路面状況変化の検知手法", *DICOMO2013* 予稿集, pp.131-138, 2013.
- [5] 高山毅, 他; "プローブパーソンによる集合を用いた路面凍結情報提供方式の提案と評価", *情報処理学会論文誌 49(7)*, pp.2549-2573, 2008.
- [6] Z. Zhuang, et al.: "Improving Energy Efficiency of Location Sensing on Smartphones.", In *Proceedings of MobiSys 2010*, pp.315-330, 2010.
- [7] M. B. Kjargaard, et al.: "EnTracked: energy-efficient robust position tracking for mobile devices", In *Proceedings of MobiSys 2009*, pp.221-234, 2009.
- [8] F. B. Abdesslem, et al.: "Less is More: Energy-Efficient Mobile Sensing with SenseLess", In *Proceedings of MobiHeld 2009*, pp.61-62, 2009.
- [9] 上坂大輔, 他; "手に保持されたセンサを用いた歩行者向けデッドレコニング手法の提案", *情報処理学会論文誌*, Vol.52 No.2, pp.558-570, 2011.
- [10] H. Hopfner, M. Schirmer.: "Energy efficient continuous location determination for pedestrian information systems", In *Proceedings of MobiDE 2012*, pp.58-65, 2012.
- [11] 伊藤嘉博, 他; "センサデータを利用した経路推薦のためのデータベースシステム", *DICOMO2011* 予稿集, pp.226-236, 2011.
- [12] 倉沢央, 他; "参加型センシングにおけるセンシング誤差を考慮したサンプリングレート制御手法", *DICOMO2013* 予稿集, pp.240-248, 2013.
- [13] ADMP401 データシートおよび製品情報, <http://www.analog.com/jp/mems-sensors/mems-microphones/admp401/products/product.html> (2013.10.07).
- [14] 温湿度センサ【SHT7x】 | センシリオン株式会社, http://www.sensirion.co.jp/doc/product/h_sel7d.html (2013.10.07).
- [15] 財団法人日本デジタル道路地図協会; "道路の区間 ID テーブル標準 ver1.0", 2011.
- [16] S. Brakatsoulas, et al.: "On Map-Matching Vehicle Tracking Data", In *Proceedings of VLDB 31st*, pp.853-864, 2005.
- [17] GALAXY NEXUS SC-04D - 概要 — SAMSUNG, <http://www.samsung.com/jp/consumer/mobilephone/smartphone/docomo/SGH-N044TSNDNM> (2014.01.10).
- [18] The Gaia-SINS federated project home-page, <http://www.gaia-gis.it/gaia-sins/> (2014.01.13).
- [19] 不動産用語「所要時間 (徒歩所要時間)」とは, <http://www.re-words.net/description/0000000059.html> (2014.01.13).
- [20] 肥田一生, 他; "ばねモデルを使った低消費電力なリアルタイム測位システム", In *Proceedings of CDS 2010*, 2010.
- [21] K. Park, et al.: "Smartphone-based pedestrian tracking in indoor corridor environments", *Personal and Ubiquitous Computing*, Vol. 17, Issue 2, pp.359-370, 2013.
- [22] S. Murata, et al.: "A wearable projector-based gait assistance system and its application for elderly people", In *Proceedings of UbiComp 2013*, pp. 143-152, 2013.
- [23] Open Street Map, <http://www.openstreetmap.org/> (2014.01.27).
- [24] 準天頂衛星システムについて: 宇宙開発 - 内閣府, <http://www8.cao.go.jp/space/qzs/qzs.html> (2014.01.29).
- [25] 村田哲史, 他; "センサの帯同場所を考慮した個人参加型センシングのための環境センサモジュールと基盤ソフトウェア", *組込みシステムシンポジウム論文集*, pp. 73-78, 2012.
- [26] 細川茂樹, 藤波香織; "傘を用いた細粒度リアルタイム雨量プローブシステムの開発", *電子情報通信学会ヒューマンプロブ研究会第 7 回研究発表会*, 2012.