

様々な状況と環境に対応できる PDR ベースの屋内位置推定ライブラリの基礎検討

外山 瑠起 梶 克彦^{†1}

概要：現代社会において、屋内位置推定技術は重要な技術である。屋内の人の動きを把握してビル内のナビゲーションなどに活用するなど様々な活用方法が考えられる。多様な状況や環境で屋内位置推定をするには、個々の条件に適した位置推定手法の組み合わせや選択が重要である。屋内位置推定の手法として PDR がある。PDR はスマートフォンなどから得られるセンサデータを元にある地点からの相対的な位置を推定する技術である。PDR はスマートフォンなどの機器さえあれば環境に左右されず一定の推定ができる手法である。一方で PDR は時間の経過に応じて特有の誤差が蓄積する問題がある。そのため PDR の誤差を環境情報などを使用して補正する必要がある。本研究の目的は様々な状況と環境に対応できる PDR ベースの屋内位置推定ライブラリの開発である

Consideration of a PDR-based indoor location estimation library for various situations and environments

TOYAMA RYUKI KAZI KATSUHIKO^{†1}

1. はじめに

屋内位置推定技術は、現代社会において重要な役割を果たしており様々な活用が期待できる。屋内位置推定技術が使用される一例として、ショッピングモール施設でのナビゲーションシステムが挙げられる [1]。このシステムでは顧客が店内で商品を探している際、その位置情報を元にしたナビゲーションシステムを通じて目的の商品が置かれている売り場まで案内するシステムである。

屋外における位置推定技術として GPS が広く利用されているが、屋内環境では建物の壁や天井が GPS 衛星からの電波を遮断してしまい、位置推定精度が大きく低下してしまう問題があり、別のアプローチが必要とされている。屋内位置推定手法として、PDR (Pedestrian Dead Reckoning) が挙げられる。PDR は主に、加速度計、ジャイロスコプ、磁気センサなどのセンサを利用して歩行者のステップ数、歩行速度、歩行方向を推定する。その情報を元に歩行

者がどのくらいの距離をどの方向に移動したかを累積的に計算して基準となる位置からの相対的な位置を推定する手法である。他の例として、Wi-Fi の電波を使用した屋内位置推定手法がある。Wi-Fi を利用した屋内位置推定は、Wi-Fi アクセスポイントからの信号強度を利用して位置推定を行う。特定の地点でのフィンガープリントを予め取得しておきそれと比較して推定を行う手法や、3つのアクセスポイントからの電波強度を利用して三角測量を行う手法などがある。

しかしこれらの手法は特定の状況や環境に特化したものであり、すべての屋内環境で同様の効果を発揮するわけではない。例として先ほどあげた Wi-Fi を利用した屋内位置推定の場合、地下施設や Wi-Fi アクセスポイントが設置が難しい場所では、信号が弱いため正確な位置推定が難しくなる。PDR の場合各種センサによる誤差が蓄積されるため、時間経過とともに誤差が大きくなってしまう問題がある。そのため長時間の屋内移動場合は精度が低下してしまう。

これらの問題を解決するためには状況や環境に応じて適切な位置推定手法の選択または組みわせる必要がある。本

¹ 愛知工業大学大学院 経営情報科学研究科

² 愛知工業大学 情報科学部

^{†1} 現在、マルチメディア、分散、協調とモバイルシンポジウム
Presently with DICOMO2023

研究の目的は様々な環境に対応できるオフライン屋内位置推定ライブラリの検討である。ここでいうオフラインとは、あらかじめ取得したデータを元に位置推定を行うという意味である。先ほどあげた長時間の屋内移動の場合、PDRで推定した位置情報をWi-Fiの電波強度を利用した位置推定でセンサの誤差を修正すれば、より正確な位置推定を行える。これらの補正を自由に組み合わせて使えるライブラリの検討を行い、様々な状況に対応できるオフライン屋内位置推定ライブラリを実現する。



図 1: 様々な状況と環境に対応できる
PDR ベースの屋内位置推定ライブラリの概要

2. 関連研究

屋内位置推定の手法において環境内に設置された機器を利用し推定する絶対位置測位手法がある。絶対位置測位は、屋内に設置された機器からの情報を元に位置を推定する手法である。例えばBluetoothやWi-Fiなどの電波を利用した手法がある。屋内に設置した近接特化型のBLEビーコン3つからの電波強度を利用して三角測量を行い位置推定を行う研究[1]やGMMを使用してWi-Fiの電波強度分布をモデル化し、それを元に位置推定を行う研究[2]がある。これらの手法は設置された機器がない場合や設置が難しい場合は使用できない問題がある。

PDRと絶対位置測位を組み合わせて屋内位置推定を行う手法がある。PDRとWi-Fiの受信強度を用いたプロキシミティベースの位置推定を行う研究[3]やBLEビーコンの受信信号強度の変異を利用した移動変異推定とPDRとの併用による累積測位誤差の補正を行う研究[4]がある。またPDRとマップマッチングを組み合わせて位置推定を行う研究[5]や歩行時の磁気データのみを用いて位置推定を行う研究[6]がある。これらの研究で示されているようにPDRと絶対位置測位を組み合わせた手法は、お互いのデメリットを補えるため屋内位置推定をする上で有用な手法である。

本研究でもPDRによる位置推定を行いその結果に対し

て、BLEビーコンの電波強度を使用した補正やマップマッチングによる補正をなどが行えるライブラリを検討する。さらに初期位置や終了位置などの様々な状況の情報で補正できるように包括的かつ柔軟な屋内位置推定ライブラリの検討を行う。

3. PDRベースの屋内位置推定ライブラリの検討

3.1 要求仕様

屋内位置推定は状況や環境によって推定に使用できる情報が異なるため、ライブラリを作る上でその具体的な例を考える必要がある。例えば、大学内や病院などのWi-Fiのアクセスポイントが多く設置されている場所ではWi-Fiの電波強度を利用した位置推定が有効である。他の例として展示会場や大きなアトリウムなどの広い開放空間が考えられる。このような場所ではWi-Fiのアクセスポイントの配置が難しく、信号のカバレッジが不均一になりやすくWi-Fiを利用した位置推定は難しい。このような場所の場合BLEビーコンを配置してその電波強度を利用した位置推定ができると考えられる。正しこのような空間で正確な位置推定を行うには、BLEビーコンの多数配置する必要がある。このような場合はPDRを利用した位置推定が有効である。PDRは歩行者の歩行速度や歩行方向を推定して位置推定を行う手法である。この手法では歩行者がスマートフォンを持っているだけで良いため、先ほどあげたような場所でもコストをかけずに位置推定を行えると考えられる。しかしPDRはセンサのわずかな誤差が、時間経過とともに蓄積され位置推定の誤差が大きくなってしまいう問題がある。そのため長時間の歩行の場合は位置推定の精度が低下してしまう。このような問題を解決する手法として[3]や[4]や方法が提案されている。

これらの手法のように基本的なベースをPDRで位置推定を行い、その結果を補正できるようなライブラリの検討を行う。ライブラリを作る上での想定環境としてXDR Challenge Track5の環境を元に考える。XDR Challengeは屋内位置推定の精度を競うコンテストである。このコンテストのTrack5では歩行者が高速道路のサービスエリアを歩行する。歩行者は腰にスマートフォンをつけた状態でLiDARと呼ばれる光を使った距離測定技術を搭載したハンドヘルドLiDAR(以下、LiDAR)を持ち施設内を歩行する。参加者にはスマートフォンから得られた、加速度、角速度、磁気センサのデータとLiDARから得られた正解の歩行軌跡の座標が提供される。施設内にはBLEビーコンが配置されており、歩行時の各ビーコンからの電波受信強度が提供される。他に与えられるデータとして、フロアマップ情報、フロアマップにおける各BLEビーコンの配置情報、歩行者の初期位置、終了位置が与えられる。本来であれば病

院や学校などのより多くの施設で利用できるライブラリにする必要がある。しかしそれら全てに対応したライブラリを初めから作成するのは難しいためまず基本的な屋内位置推定環境が整っている XDR Challenge の環境でのライブラリの基礎検討と検証を行う。

3.2 ライブラリ

関数に必要な引数の情報とその対応表を表 1 に示す。詳しい関数の説明や内部実装については後述する。引数の情報は大きく分けてセンサ情報、環境情報、その他の 3 つに分けられる。センサ情報はスマートフォンから得られる加速度、角速度、磁気センサ、BLE ビーコンの電波情報などが含まれる。環境情報はフロアマップ情報、フロアマップにおける各 BLE ビーコンの配置情報などが含まれる。これらの環境情報は全てセンサデータが与えられる前に得られる情報である。その他はセンシング中、またはセンシング前に得られる情報であり、初期位置、終了位置などの情報が該当する。

説明する関数の引数に必要な情報は全て表 1 と紐づいている。言語には Python を使用した。Python はデータ解析や機械学習などの分野で広く使われており、ライブラリを使用するユーザーにとっても比較的扱いやすい利点がある。まず基本的な PDR の処理を行う関数を Listing 1 に示す。この関数では加速度データフレーム (以下、DF)、角速度 DF を使用して位置推定を行う。加速度 DF、角速度 DF のデータフレームのカラム名とデータ型を表 2、表 3 に示す。オプション引数として正解初期座標 (ground_truth_first_point) を与えられる。正解初期座標は辞書型で表 4 に示す。戻り値は時間経過に伴う 2 次元座標の DF と角度の DF であり、それぞれのカラム名とデータ型を表 5、表 6 に示す。戻り値で角度 DF を返す理由として、次の補正処理をする際に角速度よりも扱いやすいためである。

歩幅の推定を行っている研究は多くある。機会学習を用いた研究 [7]、多変量解析を用いた研究 [8]、超音波センサーガジェットを用いた研究 [9] などがある。本関数の内部処理では歩幅の値は固定値として扱っている。本来であれば歩幅は身長、性別、年齢などの複数の要素によって動的に変化するため固定値なのはありえず、先ほど挙げた研究のように歩幅を推定する必要がある。しかし本ライブラリの目的は正確な歩幅を用いた PDR による位置推定ではない。PDR で推定した歩行軌跡を環境情報などを用いて補正を行い軌跡全体の最適化を行えるライブラリの検討である。そのため歩幅の推定は行わず固定値として扱う。また同様の理由で歩行タイミングの検出も正確には行わず、加速度の値が特定の閾値を超えた時に歩行タイミングとして扱っている。図 2 にリスト 1 を用いて PDR による位置推定を行った結果を示す。LiDAR で取得した座標をもとに出力された軌跡を図 3 に示す。これを本論では正解軌跡とする。

図 2 と図 3 を比較すると PDR による軌跡は正解軌跡と比べて大きくずれているのがわかる。PDR 特有の解決すべきものとして軌跡そのものの形状を正解軌跡に近づける問題と絶対位置との関連付けの問題がある。本ライブラリを用いてこれらの問題を解消し正解軌跡に近づけていく。

Listing 1: 基本 PDR

```
1 Axis2D = Literal["x", "y"]
2 def estimate_trajectory(
3     acc_df: pd.DataFrame,
4     gyro_df: pd.DataFrame,
5     *,
6     ground_truth_first_point: dict[Axis2D,
7     float] | None = None,
8 ) -> tuple[pd.DataFrame, pd.DataFrame]:
```

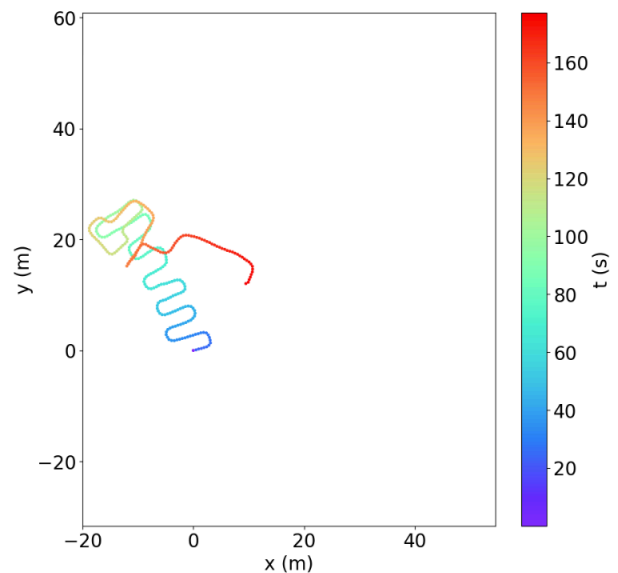


図 2: PDR による軌跡

Listing 1 に示される関数に正解初期座標を与えたのが図 4 である。予め正解座標が判明している場合は PDR による軌跡の初期位置を補正できる。

図 4 の軌跡には PDR 特有のドリフト現象が見られる。PDR では角速度から進行方向を求めてその方向を元に歩行軌跡を描く。そのため角速度センサーにわずかなでも誤差が含まれると時間経過とともにその誤差が大きくなり軌跡の形状が本来の軌跡から外れる。この問題を解決するには角速度データに含まれる累積誤差を取り除く必要がある。

Listing 2: ドリフト除去

```
1 def remove_drift_in_angle_df(
2     acc_df: pd.DataFrame,
3     angle_df: pd.DataFrame,
4     ground_truth_point_df: pd.DataFrame,
5 ) -> tuple[pd.DataFrame, pd.DataFrame]:
```

	関数名	センサ情報				環境情報				その他			
		加速度	角速度	角度	BLE ビーコン	フロアマップ	磁気	BLE ビーコン	正解初期	正解補正			
					電波強度		FP	基地局位置			FP	座標	方向
基本 PDR	estimate_trajectory	○	○							△			
角速度から角度推定	convert_to_angle_from_gyro		○										
ドリフト補正	remove_drift_in_angle	○		○						○		○	
初期方向補正 フロアマップ	rotate_trajectory_to_optimal_alignment_using_map	○		○		○				△			
初期方向補正 BLE	rotate_trajectory_to_optimal_alignment_using_ble	○	○		○			○		△			
マップマッチング補正	move_unwalkable_points_to_walkable	○	○			○							
安定歩行区間補正		○	○										
初期方向補正 BLE FP		○	○						○				

表 1: 関数に必要な情報とその対応表

注: ○ は必須引数, △ はオプション引数を示す

カラム名	単位	データ型
ts	s (秒)	float
x	m/s ²	float
y	m/s ²	float
z	m/s ²	float

表 2: 加速度 DF

カラム名	単位	データ型
ts	s (秒)	float
x	rad/s (ラジアン/秒)	float
y	rad/s (ラジアン/秒)	float
z	rad/s (ラジアン/秒)	float

表 3: 角速度 DF

	データ型	説明
key	str	x または y
value	float	座標

表 4: 正解初期座標 DICT

カラム名	単位	データ型
ts	s (秒)	float
x	m(メートル)	float
y	m(メートル)	float

表 5: 座標 DF

カラム名	単位	データ型
ts	s (秒)	float
x	rad (ラジアン)	float
y	rad (ラジアン)	float
z	rad (ラジアン)	float

表 6: 角度 DF

ドリフトを取り除く関数を Listing2 に示す. 引数として加速度 DF, 角速度 DF, 正解座標 DF を受け取る戻り値は時間経過に伴う 2 次元座標の DF と角度の DF を返す. ドリフト補正のプロセスは, ドリフトの値を動的に計算し, それを各時刻の角度データから差し引く. このドリフト補

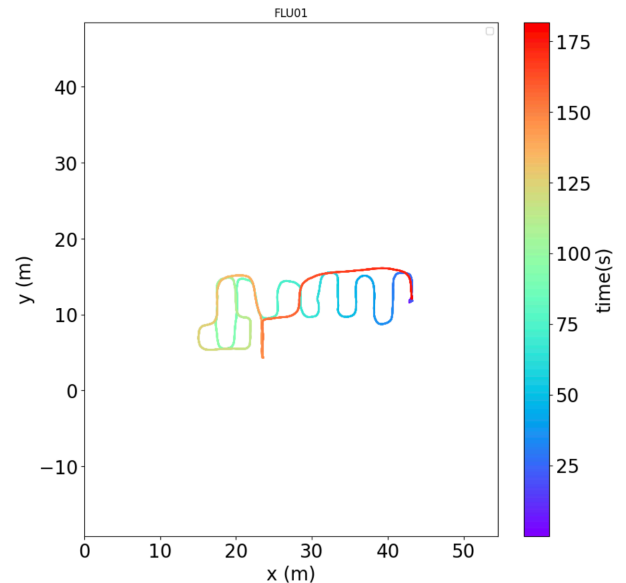


図 3: 正解軌跡

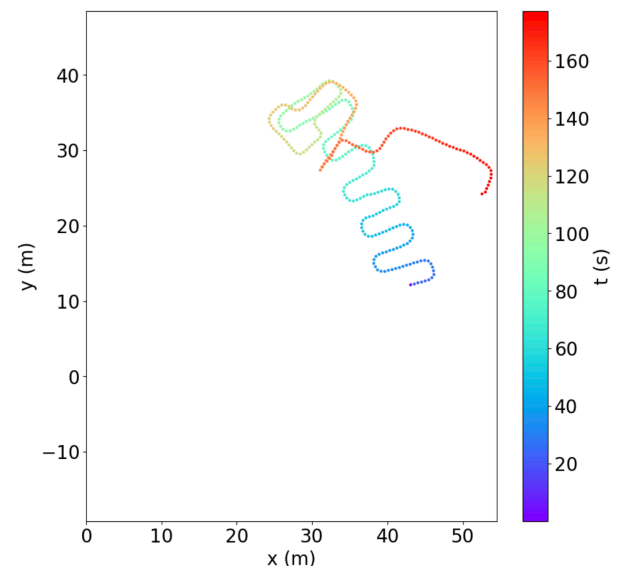


図 4: 正解初期座標が存在

正プロセスは, 式 (1) で表される. $\theta'(t)$ は時間 t における補正後の角度, $\theta(t)$ は補正前の角度, d はドリフトの大きさを意味する. この式は, 時間経過に伴うドリフトの累積効果を補正するために使用される.

$$\theta'(t) = \theta(t) - (d \times (t)) \quad (1)$$

補正の効果を評価し適切なドリフトを見つけるために、ユークリッド距離を用いて、2つの正解座標の差異を計算する。式(2)は、正解座標 (x_n, y_n) と正解座標 (x_{n+1}, y_{n+1}) との間のユークリッド距離 E を示している。この式に基づきドリフト値に対してグリッドサーチを行い距離が最小になるドリフト値を探す。最小のドリフト値を角度 DF から引きそれに基づいた座標 DF と角度 DF を返す。図5に示すように、ドリフト補正後の軌跡は、元の軌跡と比較して正解軌跡の形状に近づいている。このアルゴリズムでは正解座標 (x_n, y_n) と正解座標 (x_{n+1}, y_{n+1}) の距離が近い時に特に有効である。この処理は (x_{n+2}, y_{n+2}) など2つ以上の座標が存在する場合も同様に適用できる。

$$E = \sqrt{(x_g - x_{\text{ref}})^2 + (y_g - y_{\text{ref}})^2} \quad (2)$$

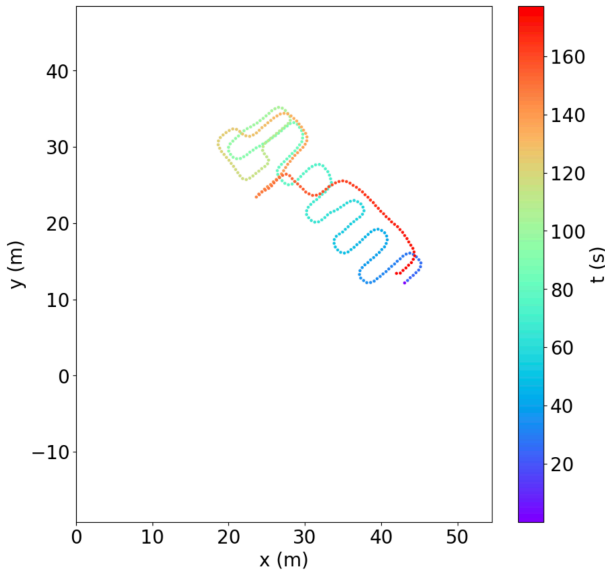


図 5: ドリフト補正後の軌跡

図5の軌跡の問題点として初期方向の誤差がある。初期方向が誤っていると、歩行者の実際の移動経路と大きく異なる軌跡になる。この問題を解決するためには、適切な初期方向を見つけて軌跡全体を回転させる必要がある。フロアマップ情報を元に軌跡を回転させる関数を Listing3 に示す。引数として加速度 DF, 角度 DF, 表7に示すフロアマップ情報 DICT, フロア名, 及びマップの 1px あたりの距離を受け取る。内部の処理としては軌跡を回転させその時の水平垂直方向の割合を計算する。軌跡における垂直成分と水平成分を可視化したものが図6である。この割合が最

も大きい回転角度をグリッドサーチを用いて探し最適な角度を見つける。しかしこの処理だけでは適切な初期方向は絞り込めない。割合が大きいものがあっても 90 度回転させるごとに水平垂直方向の割合が同一になるため4つの角度から適切な初期方向を見つける必要がある。この絞り込みの処理としてマップ上の通行可能、不可能な座標の情報を利用する。各回転角度での軌跡座標がマップ上で通行可能なポイントの数を計算し最も多いポイントを持つ回転角度を選択する。この処理を適用した結果が図7である。補正前と比べて軌跡の初期方向が正解軌跡に近づいている。

Listing 3: 初期方向補正

```
1 def rotate_trajectory_to
2     _optimal_alignment_using_map(
3     acc_df: pd.DataFrame,
4     angle_df: pd.DataFrame,
5     map_dict: dict[str, np.ndarray],
6     floor_name: str,
7     dx: float,
8     dy: float,
9     *,
10    ground_truth_first_point: dict[Axis2D,
11                                   float] | None = None,
12 ) -> tuple[pd.DataFrame, pd.DataFrame]:
```

データ型		説明
key	str	floor の名前
フロアマップの画像データ.		
value	np.ndarray	各フロアのブール値の NumPy 配列

表 7: フロマップ DICT

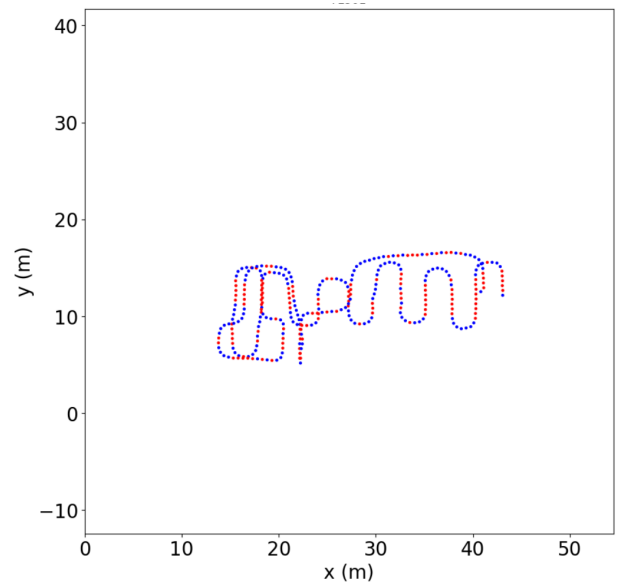


図 6: 垂直成分と水平成分の可視化

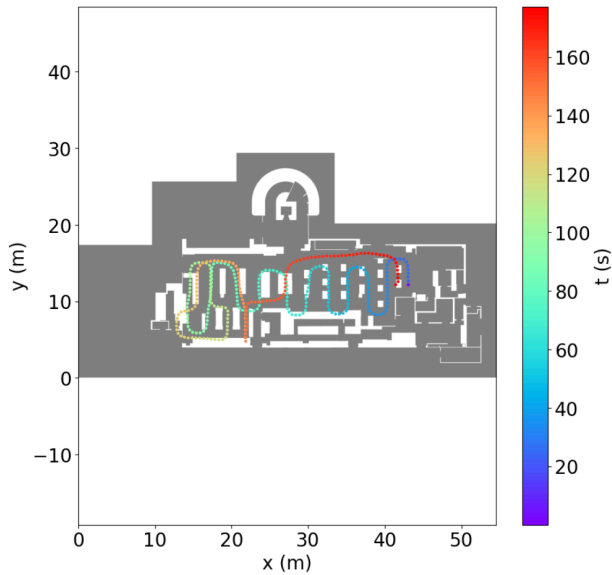


図 7: 初期方向の補正後の軌跡

フロアマップ情報を用いた初期方向補正ではマップの存在可能な点の分布によっては正しく機能しない場合がある。別の方法として BLE ビーコンの情報を用いた初期方向補正を行う関数を提供する。関数を Listing4 に示す。この関数では加速度 DF, 角度 DF, BLE ビーコンの電波強度 DF, BLE ビーコンの基地局 DF を受け取る。BLE ビーコンの電波強度 DF と BLE ビーコンの基地局 DF のカラム名とデータ型を表 6, 表 7 に示す。戻り値は時間経過に伴う 2 次元座標の DF と角度の DF を返す。一定の強い RSSI の電波を受信した際の時間情報を元に時間的に近い推定軌跡の座標を取得する。図 8 に示した図は時間的に近い推定軌跡の座標を時間経過に応じた色で表しており青色の座標が配置された BLE ビーコンの座標を表している。

推定した軌跡の受信した BLE ビーコンの基地局の座標との距離を計算する。この総和が最小となるような回転角度をグリッドサーチで探し最適な角度に補正を行う。BLE ビーコンの基地局の座標との距離を計算する。

Listing 4: BLE を使用した初期方向補正

```
1 def rotate_trajectory_to_optimal
2     _alignment_using_ble(
3         acc_df: pd.DataFrame,
4         angle_df: pd.DataFrame,
5         ble_scans_df: pd.DataFrame,
6         ble_position_df: pd.DataFrame,
7         *,
8         ground_truth_first_point: dict[Axis2D,
9             float] | None = None,
9     ) -> tuple[pd.DataFrame, pd.DataFrame]:
```

7 に示す軌跡の問題点として人間が歩行不可能領域を通過している点がある。現実の人間がこのような場所を通過しないため、このような軌跡は不適切である。そのため軌

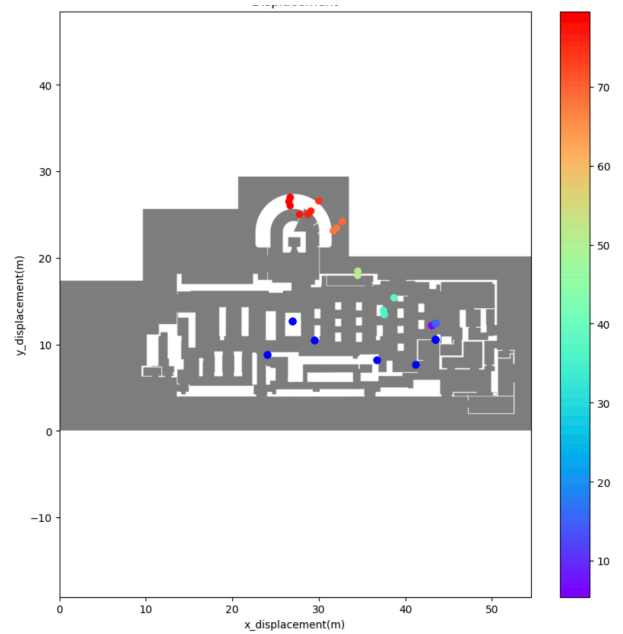


図 8: 強いビーコン電波を受信した際の時間的に近い軌跡の座標

カラム名	単位	データ型
ts	s (秒)	float
bdaddress	なし	str
rsssi	dBm	int

表 8: BLE ビーコン電波強度 DF

跡が歩行不可領域に存在する場合は、歩行可能な領域に移動させる処理が必要である。この問題を解決する処理として Listing5 に示すマップマッチング補正関数を提供する。マップマッチング補正関数は加速度 DF, 角度 DF, フロアマップ情報 DICT, フロア名, 及びマップの 1px あたりの距離を受け取る必要がある。戻り値は時間経過に伴う 2 次元座標の DF のみを返す。内部の処理の関係上補正後の角度 DF は返すのが難しいためである。関数内部ではまず加速度と角度のデータを基にして軌跡を推定する。この軌跡に対して、各地点での座標が与えられたフロアマップ上の歩行可能な領域に存在するかどうかを検証する。検証の結果、各地点での座標が歩行不可能な領域に存在する場合、当該座標から最も近い歩行可能な座標を幅優先探索アルゴリズムを用いて探す。該当する座標が見つかった場合、該当座標と当該座標以降の軌跡の座標を歩行可能な座標に平行移動して補正を行う。当該座標の補正が終了後、次の座標に対して同様の処理を行う処理を繰り返す。これによって軌跡の各地点が歩行可能な領域に存在するようになり、軌跡全体が最適化される。図 9 に示すように、マップマッチング補正後の軌跡では歩行不可能な領域に存在していた地点がが歩行可能な地点に移動されている。

Listing 5: マップマッチング補正

```

1 def move_unwalkable_points_to_walkable(
2     acc_df: pd.DataFrame,
3     angle_df: pd.DataFrame,
4     map_dict: dict[str, np.ndarray],
5     floor_name: str,
6     dx: float,
7     dy: float,
8     *,
9     ground_truth_first_point: dict[Axis2D,
10     float] | None = None,
11 ) -> pd.DataFrame:

```

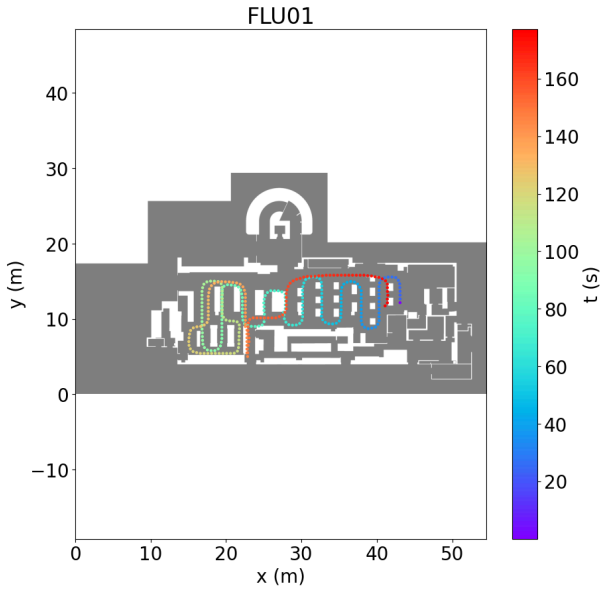


図 9: マップマッチング補正後の軌跡

カラム名	単位	データ型
bdaddress	なし	str
x	m	float
y	m	float
floor_name	なし	str

表 9: BLE ビーコン基地局 DF

4. ライブラリの検証と他環境における検討

この章では、提案したライブラリの有効性を検証する。検証は XDR Challenge での評価を元に行い、また他環境でこれらのライブラリが適用可能かを検討する。

4.1 XDR Challenge 環境での評価

XDR Challenge では PDR ベンチマーク標準化委員会によって提供された評価フレームワークを使用して評価が行われた。このフレームワークでは I_{ce} (CE: 円形誤差), I_{ca} (CA: 局所空間における円形精度), I_{eag} (誤差蓄積

勾配), I_{ve} (VE: 速度誤差), $I_{obstacle}$ (障害物回避要件) の 5 つの評価指標が用いられた。総合評価指標は式 3 に示す式で計算される。このライブラリを使って得られた評価と各指標の重みを表 10 に示す。 I_{ce} , I_{eag} , I_{ve} , $I_{obstacle}$ では一定の精度を得られた。しかし I_{ca} では精度が低かった。 I_{ca} の値が低いと場合局所空間における位置推定誤差の分布が広い。これは環境条件の変化やセンサデータの微妙な違いが、位置推定結果に大きな影響を与える可能性が高い。この問題を解決するためには、PDR アルゴリズムを改善し、より精度の高い位置推定を行う必要がある。

$$I_i = W_{ce} \times I_{ce} + W_{ca} \times I_{ca} + W_{eag} \times I_{eag} + W_{ve} \times I_{ve} + W_{obstacle} \times I_{obstacle} \quad (3)$$

指標	値 (%)	重み
I_{ce} (CE: 円形誤差)	88.55	0.25
I_{ca} (CA: 局所空間における円形精度)	62.51	0.20
I_{eag} (EAG: 誤差蓄積勾配)	93.02	0.25
I_{ve} (VE: 速度誤差)	95.55	0.15
$I_{obstacle}$ (障害物回避要件)	93.48	0.15
I (総合評価指数)	86.25	

表 10: 評価指数の概要

4.2 駅の構内での検討

駅構内での PDR での位置推定をする場合を考える。駅の場合、改札口という通過するポイントがある。通過した情報を仮に利用できるとしたら、本ライブラリで提案した補正座標を用いたドリフト補正処理を使えると考えられる。また大規模な駅構内の場合、フロアマップ情報が公開されている可能性も高い。そのためフロアマップ情報を用いた初期方向補正やマップマッチング補正も有効だと考えられる。

4.3 大学のキャンパスでの検討

大学のキャンパスでの PDR での位置推定をする場合を考える。大学のキャンパスの場合 Wi-Fi のアクセスポイントが多く設置されている。本ライブラリでは BLE ビーコンを利用した補正を行っているが BLE を Wi-Fi に置き換えれば同様の処理が可能である。設置する形式の BLE ビーコンと違いあらかじめ存在する Wi-Fi の基地局情報を把握するのはコストがかかる可能性がある。このような場合にライブラリが提供する FP を使った補正が有効だと考えられる。

5. まとめ

本論文では、環境情報などを利用した PDR ベースの位

置推定ライブラリの基礎検討を行った。補正に利用できる情報をセンサ情報、環境情報、その他の3つに分類し、それぞれの情報を用いた補正処理を提案した。その結果として、XDR Challenge 環境下では一定の精度を獲得した。また他環境においても本ライブラリが適用可能であるか検討を行った。課題としては、PDR アルゴリズムの改善や挙げられる。また利用できる情報の拡充も課題として挙げられる。

参考文献

- [1] 酒井瑞樹, 森田裕之. Bluetooth を用いた屋内位置推定手法の提案. 経営情報学会 全国研究発表大会要旨集 2016 年秋季全国研究発表大会, pp. 53–56. 一般社団法人 経営情報学会, 2016.
- [2] 浅見宗広, 福井隆司, 葛城友香, 大杉苑子ほか. Gmm を用いた wifi 位置推定手法の実用化に向けた課題解決. 研究報告高度交通システム (ITS), Vol. 2013, No. 2, pp. 1–8, 2013.
- [3] 田巻櫻子, 田中敏幸. Wi-fi および端末センサ情報を用いた 3 次元屋内位置測位手法の検討. 国際 ICT 利用研究学会論文誌, Vol. 2, No. 1, pp. 24–30, 2018.
- [4] 工藤大希, 堀川三好, 古舘達也, 岡本東ほか. Ble 測位および pdr を用いたハイブリッド型屋内測位手法の提案. 第 79 回全国大会講演論文集, Vol. 2017, No. 1, pp. 325–326, 2017.
- [5] 吉見駿, 村尾和哉, 望月祐洋, 西尾信彦ほか. マップマッチングを用いた pdr 軌跡補正. 研究報告ユビキタスコンピューティングシステム (UBI), Vol. 2014, No. 20, pp. 1–8, 2014.
- [6] 村田雄哉, 梶克彦, 廣井慧, 河口信夫, 神山剛, 太田賢, 稲村浩ほか. 歩行時の磁気センシングデータを利用した屋内位置推定手法. 情報処理学会論文誌, Vol. 58, No. 1, pp. 57–67, 2017.
- [7] 三宅孝幸ほか. デッドレコニング開始前のセンサデータを自動学習に用いた屋内歩幅推定手法の検討. 第 74 回全国大会講演論文集, Vol. 2012, No. 1, pp. 443–444, 2012.
- [8] 根岸拓郎, 藤田悟ほか. 携帯端末のセンサ値を用いた多変量解析による歩幅推定. 第 77 回全国大会講演論文集, Vol. 2015, No. 1, pp. 299–300, 2015.
- [9] 柏本幸俊, 荒川豊, 安本慶一ほか. デッドレコニングの高精度化に向けた超音波による歩幅推定法. 研究報告モバイルコンピューティングとユビキタス通信 (MBL), Vol. 2015, No. 56, pp. 1–6, 2015.