

トラッキング画像によるパーティクルフィルタの学習法の提案

北見工業大学 ○佐藤 良介 岩館 健司 鈴木 育男 渡辺 美知子

要 旨

自己位置の推定手法としてパーティクルフィルタという手法がある. 本論文ではパーティクルフィルタを二輪ロボットに実装することを検討する. トラッキング画像をもとに, パーティクルフィルタの学習手法を提案する.

1. はじめに

ロボットが特定のタスクを実行するには, 自己位置を知ることが重要である. 自己位置推定で一般的に知られている手法として, オドメトリという手法があるが, センサデータの不確実性による誤差の累積や, 車輪の空転などの環境変化による誤差が発生する問題がある. そのため, 近年では, 自己位置を確率的に推定する手法が提案され, 実機による実証実験も行われている. そこで, 本研究においても, 自己位置を確率的な粒子として表現し, 自己位置推定を行うパーティクルフィルタによる手法を採用する.

パーティクルフィルタの手法では, 観測データに基づき設定される尤度関数によって各パーティクルの重み(重要度)を決定する必要がある. その際, ロボットに搭載されたレーザーレンジファインダーによる距離計測や, カメラ画像による自己位置の特定を行い, 尤度関数の設定を行う. しかしながら, 本研究では, 図1のような小型ロボットを用いるため, 上記のような装置をロボットに設置できない. そのため, 外部からの画像を使用することで尤度関数を求めることを提案する.

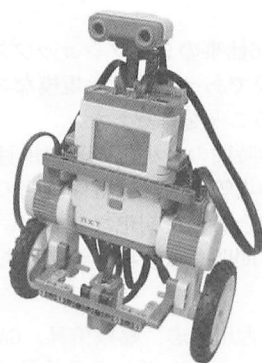


図1: 二輪倒立ロボット

2. 自己位置推定法

2.1 オドメトリ

オドメトリは, デッドレコニングとも呼ばれ, 車輪の回転角から現在位置を推定する手法である.

今想定する車体モデルを本体に車輪を二つ付けたシンプルなもの(図2)とし, そのタイヤの回転からロボットの現在の自己位置を求める.

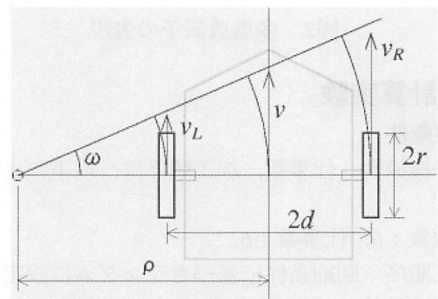


図2: 車体モデル

オドメトリの計算では, はじめに両輪の速度から旋回中心周りの運動を考え, ロボットの旋回角速度: ω , ロボット中心速度: V , 旋回半径: ρ を求める.

次にロボットが θ 回転していた場合について考え, 時間 Δt で左右の両輪がそれぞれ移動したとして, 旋回中心周りの単位時間当たりの回転角: $\Delta\theta$ を求める.

最後に, ロボットの位置, 姿勢を制御周期 i のときに x_i, y_i, θ_i として, 次の周期での位置及び角度を $x_{i+1}, y_{i+1}, \theta_{i+1}$ としたとき次式で求められる. ただし, ΔL をそのまま代用して計算すると, $\Delta\theta$ の値が大きくなった場合の誤差が生まれてくるため, $\Delta L'$ の値を用いるものとする.

$$x_{i+1} = x_i + \Delta L' \cos(\theta_i + \Delta\theta/2) \quad \dots (1)$$

$$y_{i+1} = y_i + \Delta L' \sin(\theta_i + \Delta\theta/2) \quad \dots (2)$$

以上のように計算すると, 「 Δt の間, 円弧運動をしている」としたとき, 正確な演算結果が得られる.

予備実験として, 物理エンジン(physics)を使用したシミュレーション実験を行った. ロボットモデルの各パラメータを表1にまとめる.

表1: ロボットモデルの各パラメータ

本体長さ(縦, 横, 高さ):	1.5, 1.0, 0.8
タイヤ半径:	0.5

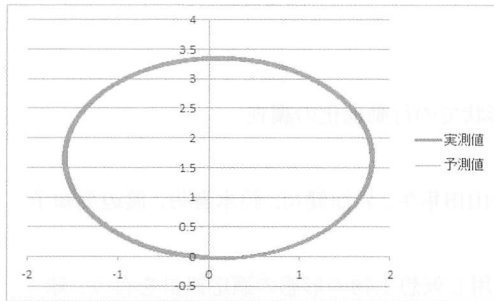


図 3: オドメトリ(円運動)

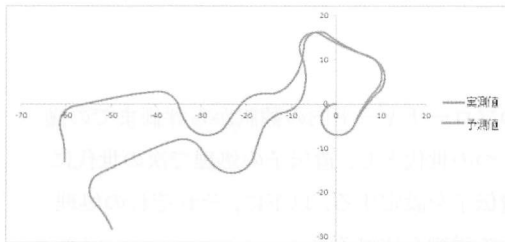


図 4: オドメトリ(軌道追跡走行)

図 3, 4 はシミュレーション実験の結果である。図 3 は円周運動における、オドメトリによる推定位置とシミュレーションによる動作の実測値を表している。この環境においては、オドメトリの推定値と実測値に差はなく、オドメトリによる自己位置推定がうまく動作していることがわかる。一方、図 4 は、任意の軌道追跡走行を行わせた結果である。急な旋回を行うなどした後に、推定値が実際の位置から少しずつ始め、誤差が蓄積され、推定位置と実測値が大きく異なっていることが見られる。このように、オドメトリの手法では長距離の移動になるほど誤差が大きくなるので、それを修正するための方法が必要になる。

2.2 パーティクルフィルタによる位置推定

パーティクルフィルタとは、時系列フィルタの一種で、主に最適解を出す際に用いられる。

i 番目のパーティクル $p_i^{(i)}$ は、ロボット位置 $r_i^{(i)} = (x_i^{(i)}, y_i^{(i)}, \theta_i^{(i)})$ と重み $w_i^{(i)}$ をもつ。

以下に位置推定の手順を示す。

(1) サンプリング

i 個のパーティクルを生成する。その際に乱数 $x\varepsilon_i^{(i)}, y\varepsilon_i^{(i)}, \theta\varepsilon_i^{(i)}$ を発生させ、各パーティクルの位置に誤差として加える。

$$r_i^{(i)} = (x_i^{(i)} + x\varepsilon_i^{(i)}, y_i^{(i)} + y\varepsilon_i^{(i)}, \theta_i^{(i)} + \theta\varepsilon_i^{(i)})$$

(2) 重み計算

各パーティクルに対し、観測データとの一致度から尤度を計算し、重みとする。重みは合計が 1 になるように正規化する。

(3) リサンプリング

各パーティクルを Δt 進め、各 $p_i^{(i)}$ に対し、 $w_i^{(i)}$ に比例した個数パーティクルを複製する。なお、複製はルーレット選択で選ばれる。複製のパラメータ x, y, θ の値に誤差を発生させる。作成されたパーティクルは、時刻 $t+1$ のパーティクルとなる。

その際にリサンプリングされたパーティクルの位置の平均を現在の位置の推定値とする。

パーティクルフィルタによる自己位置推定では、各パーティクルが実際のロボットをマネしたものと考え、重みの計算はロボットの実測値との誤差が小さいほど大きくなり、重みの大きなパーティクル周辺により多くのパーティクルがリサンプリングされ、そのパーティクルの密度により、ロボットの位置が推定される。

3. 提案手法

本研究では、図 1 の小型ロボットを用いてパーティクルフィルタによる自己位置推定を行う。ロボットは、環境中のライン上を移動するものとする。ただし、より高速な移動することを目的とするため、ライントレースにおけるライン情報はあくまで観測値として利用する。

パーティクルフィルタによる自己位置推定では、尤度関数が重要な役割を担っている。そのため、観測データと、マップ情報から尤もらしい自己位置を特定するためのデータベースが必要となる。本研究では、観測データとマップ情報のマッチングに、外部カメラから取得されるトラッキング画像を利用することを提案する。

4. おわりに

本稿では、自己位置推定に一般的に利用されるオドメトリについて、シミュレーション実験を通じて、誤差蓄積の問題点を確認した。また、パーティクルフィルタによる自己位置推定に必要な、尤度関数の設定に外部カメラからのトラッキング画像を利用する方法を提案した。現在、提案手法に必要なトラッキングシステムを作成中であり、パーティクルフィルタ実装を行う予定である。講演でその結果を公表する。

参考文献

- [1] 友納 正裕, 「移動ロボットのための確率的な事故位置推定と地図構築」, 日本ロボット学会誌 vol29 No.5, pp.423~426, 2011
- [2] 「車輪移動ロボット」, http://www.mech.tohoku-gakuin.ac.jp/rde/contents/cou_rse/robotics/wheelrobot.html