令和 3 年度 卒業論文
各パーティクルごとに観測範囲を可変にした
モンテカルロ自己位置推定

池邉 龍宏
Chiba Institute of Technology

202x 年 x 月 x 日

謝辞

本稿を執筆するにあたって助言を頂いた上田隆一准教授および上田研究室のみなさまに感謝します。

目次

第 1 章

序論

1.1 研究背景

移動ロボットが自己位置を推定する方法として確率的な位置推定法であるパーティクルフィルターがよく用いられる。用いられている例として、ros のナビゲーションスタックである amcl[1] や千葉工業大学未来ロボティクス学科の自律移動ロボットチームが開発した emcl[2] がある。パーティクルフィルターは現在の状態から推定される次のロボットの状態を多数のパーティクルに見立て、全パーティクルの尤度に基づいた重みつき平均を次の状態として近似することで自己位置を推定するアルゴリズムである。

安定的に自己位置推定を行うことは、移動ロボットがある目標へと自律移動を行うために重要である。パーティクルフィルターにおいて何らかの理由で全パーティクルが高い尤度を得られなかった場合、自己位置を見失ってしまうことがある。自己位置を見失ったまま行動をし続けてしまった場合、ロボットが走行できない状態になってしまう。そういった自己位置を見失ってしまった状態に対しての回復策として様々なリセット法がある。例えば、amcl は前回より尤度平均が小さくなるとランダムパーティクルを注入する。emcl は尤度平均がある任意の閾値より小さくなった場合、任意の範囲内にパーティクルを配置をする [3]。それぞれのリセットはいずれも尤度がある値を下回った時に掛かり、パーティクルを真値周辺に近づけることができる。

しかし、効果的なリセットではあるが未知障害物を観測した場合、不要なリセットが何度も掛かってしまうことで逆に自己位置推定に悪影響を与えることがある。例えば、既知のマップに対して未知障害物である車や人を含んだ観測情報の全てを尤度計算に用いた場合、全パーティクルの尤度が小さくなる。また、未知障害物への対策を講じていなければ、未知障害物が観測情報として入力されている間は常に全パーティクルの尤度が小さくなる。そのため、全パーティクルの尤度が、ある閾値より小さい場合はリセットが掛かり続ける。リセットが掛かりるづけるとパーティクルは収束しないので誘拐されるか自己位置推定を誤ってしまう。

安定的に自己位置推定を行うためには未知障害物に対して対応するアルゴリズムが必要である。未知障害物によって不要なリセットが掛かり続けなければ、より安定した自己位

置推定を行うことができる。

そこで本稿では各パーティクルごとに観測範囲を可変にすることで未知障害物に対して対応する方法を提案する。未知障害物が含まれていない観測範囲が与えられたパーティクルであれば、そのパーティクルの尤度は高くなる。尤度計算を繰り返し行うことで最適な観測範囲が収束し求められる。この方法は、観測範囲を変更するだけなので計算量を削減することができ、よりロバストな自己位置推定を行う事ができる。

この方法については@@@章で話す。

1.2　従来研究

1.3　研究目的

1.4　本論文の構成

第 2 章

研究の目的

そこで、上田の研究をもっと時代におもねった方法に変える手法の研究を行う。

第 3 章

各パーティクルに観測範囲を可変にした未知障害物対策

第 4 章

シミュレーション実験

第 5 章

実機実験

付録 A Raspberry Pi Cat

5.1 Raspberry Pi Cat

Raspberry Pi Cat(以下ラズパイキャット) は図 5.1 で示された二輪の差動駆動型ロボットである．また, ラズパイキャットは既製品の小型移動ロボットである．このロボットについて説明をする．

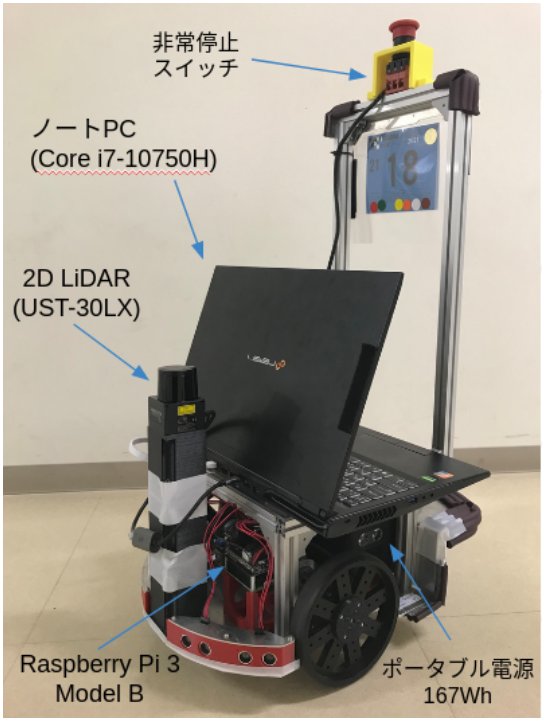

図 5.1

5.1.1 使用したコンピュータのスペック

Raspberry Pi 3 Model B

SoC は Broadcom BCM2837 1.2GHz × 4(CPU). メモリは 1GB.

ノート PC

CPU は Core i7-10750H 2.6GHz × 12. GPU は NVIDIA GeForce RTX 2070. メモリは 32GB.

5.1.2 2D LiDAR

使用した 2D LiDAR は HOKUYO の UST-30LX である. UST-30LX の走査角度は 270 度, 角度分解能は 0.25 度, 測定分解能は 1mm, 最大検出距離は 60m である.

地面などのノイズを観測せずに多くのランドマークを獲得するために図 5.2 のように地面から 36cm の高さに取り付けた.

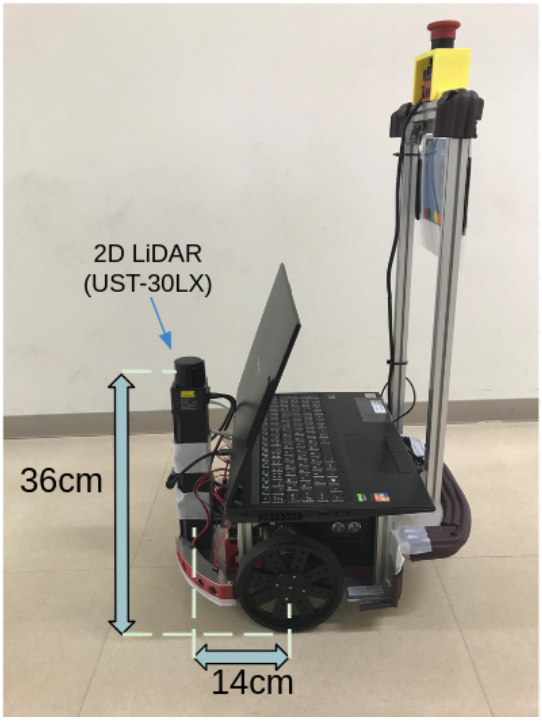

図 5.2

5.1.3 ハードウェア構成

ラズパイキャットのハードウェア構成を図 5.3 に示す. 搭載されている外界センサは, 2D LiDAR(UST-30LX) のみである. この 2D LiDAR のみで自己位置推定や障害物回避を実現している. モータの速度制御として Raspberry Pi 3 Model B が搭載されており, モータは Raspberry Pi 3 Model B から送られてくる速度指令値に基づいてモータドライバによって駆動されている.

自己位置推定や行動計画のための計算機としてノート PC(Intel i7-10750H) を搭載して

いる．また, Raspberry Pi 3 Model B とノート PC は LAN ケーブルによる接続で通信を行っている．

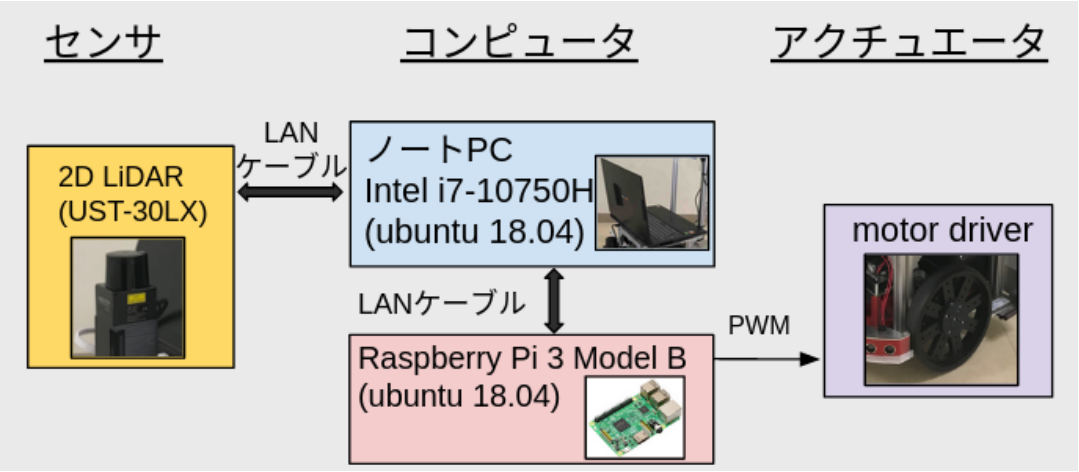

図 5.3

5.1.4 ソフトウェア構成

ラズパイキャットのソフトウェア構成を図 5.4 に示す．ラズパイキャットはロボットの制御システムとして ROS を使用している.Raspberry Pi 3 Model B では motors ノードのみが立ち上がっており, モータの制御を行っている．ノート PC では, amcl や move_base 等のノードが立ち上がっており, 自己位置推定や行動計画を行っている．

オドメトリには imu などは使用しておらず, motors ノードで速度指令値 (/cmd_vel) を積算させたデッドレコニング (/odom) を使用している．そのため, 急回転には弱いが非常に簡素なシステムになっているためシステムの立ち上げミスが少ない．

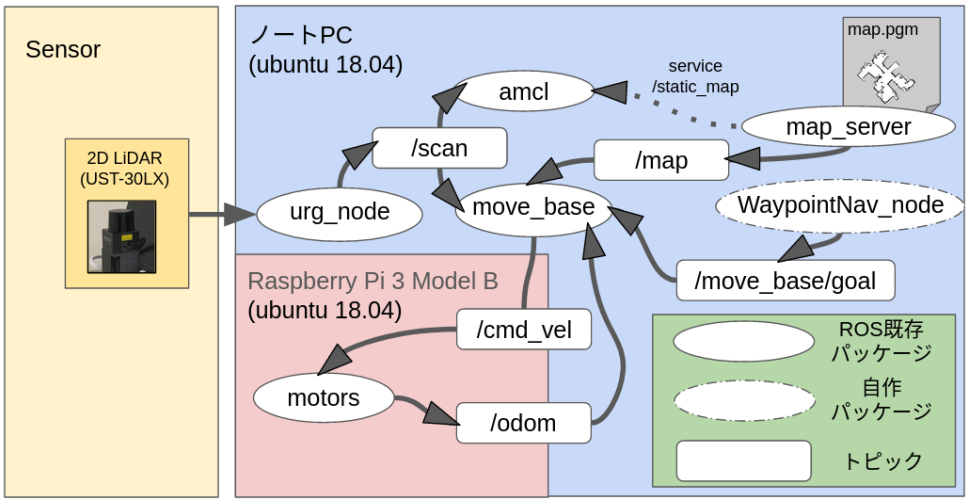

図 5.4

5.1.5 アルミフレームによる高さ増し

つくばチャレンジでのロボットの仕様条件を満たすために図 5.5 のようにアルミフレームを追加し, 高さ増しを行った. 元々の Raspberry Pi Cat の高さは 21cm であったが高さ増しを行うことによって 80cm になった。

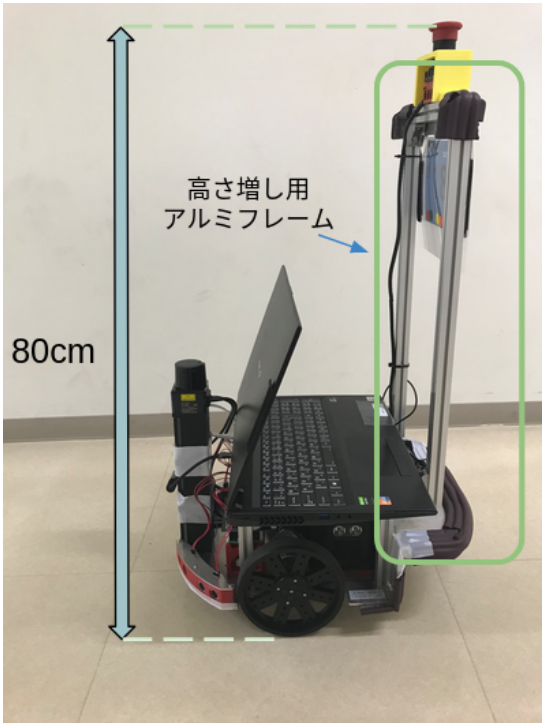

図 5.5

参考文献

[1] Brian P. Gerkey. amcl. https://github.com/ros-planning/navigation/tree/noetic-devel/amcl. (visited on 2021-12-12).

[2] Ryuichi Ueda. emcl. https://github.com/ryuichiueda/emcl. (visited on 2021-12-12).

[3] Ryuichi Ueda, Tamio Arai, Kohei Sakamoto, Toshifumi Kikuchi, and Shogo Kamiya. Expansion Resetting for Recovery from Fatal Error in Monte Carlo Localization – Comparison with Sensor Resetting Methods. In *Proc. of IROS*, pages 2481–2486, 2004.