A-16

デプスカメラを用いた3次元モンテカルロ自己位置推定の研究

北海道大学　○畠山 龍，　金井 理，　伊達 宏昭

要旨

本論文では，レーザースキャナから得られた詳細な3次元モデルを環境情報として利用し，デプスカメラをセンサとして用いた3次元自己位置推定手法を提案する．提案手法ではパーティクルフィルタに基づいたモンテカルロ自己位置推定アルゴリズムを用い，尤度計算には3次元モデルに対するGPUシーンシミュレーション手法を用いている．本提案手法により自己位置推定を行い，精度評価を行った結果，精度が既存手法より向上したことが確認できた．

1. はじめに

近年，Building Information Model(BIM)の普及や地上設置型のレーザースキャナの進歩から，屋内環境の3次元モデルの利用が容易になってきている．将来的には人の屋内ナビゲーションサービスへの利用や，BIMを現実の環境に投影するサービスを可能にするなどの利用も考えられる．それらを実現するには，人間が可搬な3次元自己位置推定システムが必要不可欠である．しかし，現状の自己位置推定の研究は，ロボットを対象とした2次元の自己位置推定に関するものがほとんどであり[1]，3次元の位置・姿勢の推定を扱ったものは少ない．その理由としては，環境地図情報として3次元モデルが必要であること，環境を実時間で3次元計測可能なセンサが必要であること，次元が増えることにより扱う状態変数が増加し，計算が複雑で膨大になることの3点が挙げられる．

そこで本研究では，SLAMで作成された環境地図内でGPUを用いて3次元自己位置推定を行った先行研究[2]を参考に，地上型レーザースキャナで作られた詳細な3次元メッシュモデルを環境地図として利用し，さらにセンサとしてデプスカメラを用いることにより，3次元自己位置推定の高精度化を試みた結果を報告する．

2. モンテカルロ自己位置推定アルゴリズム

図1に，提案する自己位置推定アルゴリズムの流れを示す．本手法は文献[2]と同様に，パーティクルフィルタに基づく状態予測を行う．パーティクルフィルタは，マルコフ連鎖確率密度分布モンテカルロ法(MCMC)を用いたシミュレーションに基づくモデルの状態推定法のひとつであり，状態の確率密度分布を状態空間に散らばる重みつきパーティクルの集合で表現し，重みに基づいたリサンプリング(A-3)，予測による伝搬(A-1)，尤度計算と重み付け(A-2)を繰り返すことによりパーティクル集合を更新する手法である．各パーティクルは状態の候補を表す．

3. 本研究における状態予測アルゴリズム

本研究ではデプスカメラの位置・姿勢の6自由度を状態量として推定する．状態推定は，以下の手順で行う．

3.1 初期分布の作成

x, y, zをモデル座標系におけるデプスカメラの位置，φ, θ, ψをカメラのロール，ピッチ，ヨー角とすると，推定すべき状態ベクトル$\boldsymbol{\mathcal{X}}$は式(1)で表される．さらに時刻tにおけるパーティクル集合P_tを式(2)のように表す．

$$\boldsymbol{\mathcal{X}} = [x, y, z, \varphi, \theta, \psi] \tag{1}$$

$$P_t = \{\boldsymbol{p}_t^1, \boldsymbol{p}_t^2, \cdots, \boldsymbol{p}_t^{N-1}, \boldsymbol{p}_t^N\} \tag{2}$$

ここで，Nはパーティクル数，$\boldsymbol{p}_t^i$は時刻tにおけるパーティクルiが持つ状態ベクトルを意味する．本研究では，初期位置(x_0, y_0, z_0)，姿勢$(\varphi_0, \theta_0, \psi_0)$をあらかじめ与える．初期状態ベクトル$\boldsymbol{\mathcal{X}}_0$は式(3)で表される．また，初期パーティクル集合$P_0$はすべてのパーティクルが同じ初期状態ベクトル$\boldsymbol{\mathcal{X}}_0$をもつ様，式(4)として定義する．

$$\boldsymbol{\mathcal{X}}_0 = [x_0, y_0, z_0, \varphi_0, \theta_0, \psi_0] \tag{3}$$

$$P_0 = \{\boldsymbol{\mathcal{X}}_0, \cdots, \boldsymbol{\mathcal{X}}_0\} \tag{4}$$

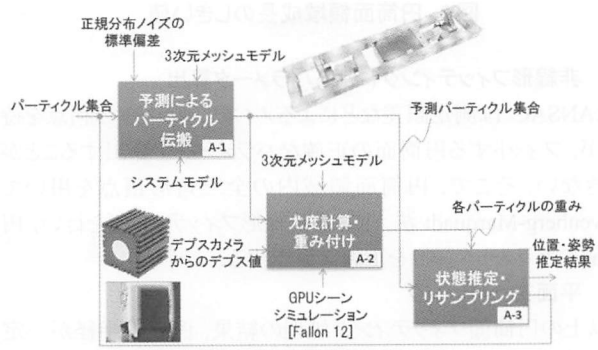

図1 提案する自己位置推定アルゴリズム

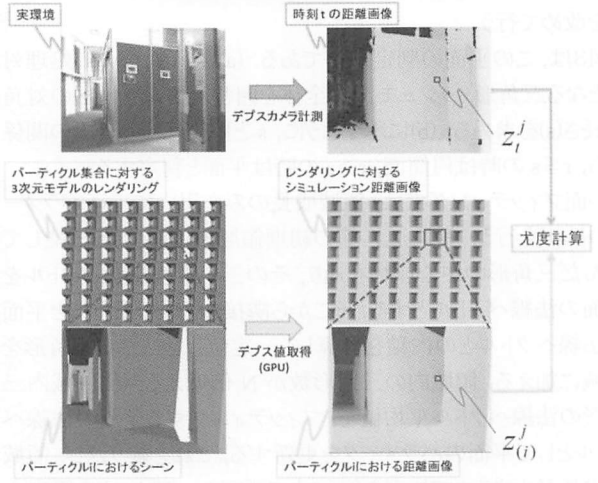

図2 GPUシーンシミュレーション[2]概要

3.2 予測によるパーティクルの伝搬(A-1)

パーティクル集合の更新のために，まずパーティクルを伝搬させる．本研究では正規分布に基づき時刻tにおけるシステムノイズ$\boldsymbol{\mathcal{V}}_t$を生成し，式(5)に示すシステムモデルにより直前のパーティクル集合P_{t-1}に含まれる各パーティクルを伝搬させる．なお，システムノイズの標準偏差ベクトル$\boldsymbol{\sigma}$は各状態変数に対してあらかじめユーザが個別に設定した．

$$\boldsymbol{\mathcal{X}}_t = \boldsymbol{\mathcal{X}}_{t-1} + \boldsymbol{\mathcal{V}}_t \qquad \boldsymbol{\mathcal{V}}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\sigma}^2) \tag{5}$$

3.3 尤度計算・重み付け(A-2)

次に，現時刻tのデプスカメラから得られた距離値集合と，3.2節でパーティクル集合P_{t-1}を伝搬させて得られた予測パーティクル集合$P_{\hat{t}}$の各パーティクルに基づいてGPUシーンシミュレーションで得られた距離値集合を比較することで，尤度を計算する．デプスカメラからの距離値の取得，GPUシーンシミュレーション，尤度計算の概要を次に述べる．

3.3.1 GPUシーンシミュレーション[2]

GPUシーンシミュレーションの概要を図2に示す．3次元モデルに対するシーンシミュレーションでは，まず，パーティクルの持つ状態を仮想的なカメラの位置・姿勢として，3次元モデルをレンダリングする．レンダリングされたモデルデ

ータにおいて，描画された各画素が持つZバッファ上のモデルまでの距離値(Zバッファ値)は，GPUにより高速に取得できる．このZバッファ値をそのパーティクルの状態におけるデプスカメラの計測データの擬似的な再現とする．このようにして，パーティクルiの仮想カメラ位置姿勢から得られた距離値集合$Z_{(i)}$を式(6)のように表す．なお，mは実際のデプスカメラの画素数と一致させる．

$$Z_{(i)} = \{z_{(i)}^1, z_{(i)}^2, \cdots, z_{(i)}^{m-1}, z_{(i)}^m\} \tag{6}$$

3.3.2 デプスカメラのデータ処理

デプスカメラからは，カメラ座標系において実際の物体表面のx, y, z座標が画素ごとに得られる．このカメラ座標系とシミュレーション上での仮想カメラ座標系を一致させ，デプスカメラから得られた点群もGPUでレンダリングすることで，3.3.1節と同様に画素ごとの実距離値を得る．このようにして，時刻tにおいてデプスカメラから得られた距離値集合Z_tを式(7)のように表す．

$$Z_t = \{z_t^1, z_t^2, \cdots, z_t^{m-1}, z_t^m\} \tag{7}$$

3.3.3 尤度計算

式(6),(7)の距離値集合$Z_{(i)}, Z_t$と，任意に設定するパラメータαから，パーティクルiの尤度$l_{(i)}$を式(8)により計算し，その中で最小のものをl_{min}，最大のものをl_{max}として，式(9)により正規化尤度$\tilde{l}_{(i)}$を計算する．

$$l_{(i)} = \sum_{j=1}^{m} \left(1 - \frac{\left| z_t^j - z_{(i)}^j \right|}{\alpha + \left| z_t^j - z_{(i)}^j \right|} \right) \tag{8}$$

$$\tilde{l}_{(i)} = \frac{l_{(i)} - l_{min}}{l_{max} - l_{min}} \tag{9}$$

3.3.4 重み付け

正規化尤度$\tilde{l}_{(i)}$を用いてパーティクルiの正規化重み$\tilde{w}_{(i)}$を式(10),(11)により求める．なお，βは任意に設定するパラメータである．

$$w_{(i)} = exp\left[-\frac{\left(\tilde{l}_{(i)} - 1\right)^2}{\beta^2} \right] \tag{10}$$

$$\tilde{w}_{(i)} = \frac{w_{(i)}}{\sum_{i=1}^{N} w_{(i)}} \tag{11}$$

3.4 状態推定・リサンプリング(A-3)

重み$\tilde{w}_{(i)}$と，状態ベクトル$\boldsymbol{p}_{\hat{t}}^i \in P_{\hat{t}}$から，式(12)の重み付き和として最終的に得られる時刻tの状態ベクトル$\tilde{\boldsymbol{x}}_t$を求める．

$$\tilde{\boldsymbol{x}}_t = \sum_{i=1}^{N} (\tilde{w}_{(i)} \boldsymbol{p}_{\hat{t}}^i) \tag{12}$$

最後に，$P_{\hat{t}}$内のパーティクルの重みに応じて，パーティクルの複製と消滅を行い，確率密度分布を近似する新たなパーティクル集合を得るリサンプリングを行う．リサンプリング後の新たなパーティクル集合をP_tとし，3.2節の処理に戻る．

4. 実験条件と結果

- 実施環境：本研究科棟4F廊下部分
- デプスカメラ：SR4000　(解像度：176×144)
- 実験：位置x, yとヨー角ψの推定　(パーティクル数：200)
- パラメータ：$\alpha = 0.1, \beta = 0.01$

センサとして利用したSR4000は**図3**のようにワゴン上に設置した．なお，実験において推定する状態変数以外は初期設定値から変化しないとする．また，各パラメータは予備実験より，試行錯誤的に決定した．計測の模式図を**図4**に示す．

1フレームあたりの自己位置推定の処理時間平均は，125msとなった．精度検証は自己位置推定によって得られた

図3 実験装置

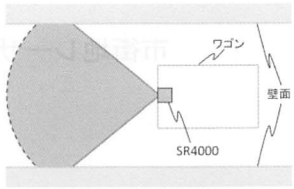

図4 計測模式図

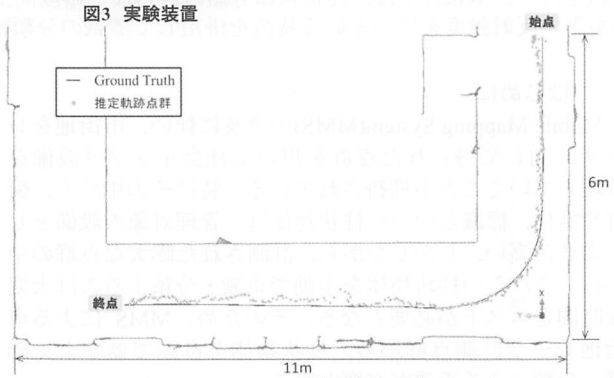

図5 Ground Truthと推定位置のプロット

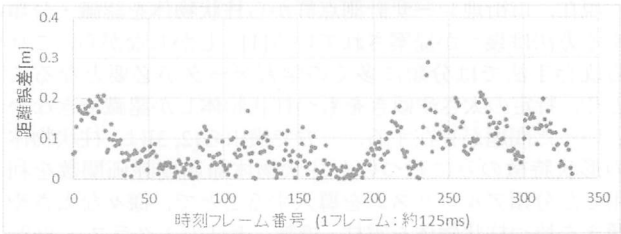

図6 各時刻フレームにおける推定位置の誤差

表1 提案手法と既存手法[2]の誤差比較

	中央値 [m]	σ [m]	3σ% [%]
提案手法	0.07	0.10	99
既存手法[2]	0.30	n/a	90

推定軌跡点群から，モデル座標系でのカメラの水平面上の軌跡点群(Ground Truth)内の最近傍点間距離を計算することで行った．Ground Truthはワゴンにマーカーを取り付け，センサの真下の床に軌跡線を描き，地上設置型レーザースキャナにより軌跡線を含めた廊下点群を計測し，環境情報として利用したモデル点群とレジストレーション(処理後の平均点間距離：約10mm)を行うことで生成した．実験におけるGround Truthと推定軌跡のプロットを**図5**，各時刻フレームにおける推定位置とGround Truth上の最近傍点間の距離の変化を**図6**に示す．また，この誤差距離値の中央値と3σ区間に含まれる割合を，Fallonらの研究[2]における類似条件下での推定結果と比較したものを**表1**に示す．

これらの結果より，3次元の自己位置推定を扱った先行研究[2]と比較し，誤差中央値が約1/4に減少し，3σ%も向上するなど，良好な結果が得られた．

5. おわりに

本研究では，地上設置型レーザースキャナの計測データから構築された高精度な3次元環境モデルを用いて，デプスカメラによる3次元自己位置推定を行う手法を提案し，実際に先行研究より推定精度が改善されることを確認した．しかし，先行研究と同じく，完全3次元自己位置推定(同時6状態変数推定)はいまだ実現できていないため，今後の課題とする．

参考文献

[1] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun, "Monte Carlo Lolalization for Mobile Robots", Proc.ICRA, Vol.2, pp.1322-1328, (1999)

[2] Maurice F. Fallon, Hordur Johannsson, and John J. Leonard, "Efficient Scene Simulation for Robust Monte Carlo Localization using an RGB-D Camera", Proc.ICRA, pp.1663-1670, (2012)