SLAM 実行時のエッジへの動的メモリオフロード制御の提案と設計

長濱幸輝[1]　石綿陽一[2]　大川猛[3]　菅谷みどり[1]

概要：近年，自律移動型ロボット活用が進んでいる．自律移動型ロボットでは，自己位置推定と環境地図作成を同時に行う SLAM 技術が広く採用されている．一方で，SLAM は各種の画像処理と演算処理の負荷が高く，組込みなどのハードウェアリソースが非常に限られているシステムでは，実行速度の低下が発生しやすいという課題がある．オフロード手法は各種提案されているが，ロボット向けミドルウェア ROS を用いた SLAM アプリに適用されておらず，またこの環境におけるロボットでの検証例は少ない．そこで，本研究では，SLAM 処理に伴う一定期間のシステムリソース情報をもとに予測モデルの構築を行い，必要に応じてメモリ上のデータをエッジに対してオフロード処理する，リソース使用予測に基づく負荷軽減手法を提案する．本稿においては，リソース使用予測に使用する予測モデルを提案し，評価を行った．また，これと併せて ROS/ROS2 と Kubernetes を組み合わせたオフロード処理のシステムの設計について検討した．

キーワード：省メモリ，ロボット，オフロード，資源利用予測モデル，ROS/ROS2，Kubernetes

1. はじめに

近年，日本では「世界一のロボット利活用社会」として，自律移動型ロボット活用が進んでいる[1].

自律移動ロボットでは，自己位置推定と環境地図作成を同時に行う SLAM (Simultaneous Localization and Mapping) 技術が広く採用されている[2]．SLAM 技術により自律移動ロボットは，自己位置や，周辺情報を把握することができることから，本技術は自律移動を行うシステムには不可欠な技術となっている．しかし，SLAM は，センサから物理情報を取得し，各種の高度な演算処理が必要であることから計算機負荷が高い．このことから，ハードウェアリソースが非常に限られているシステムでは実行速度の大幅な低下が懸念されている[3].

こうした課題に対して，Benjamin Sugar らは，SLAM の計算負荷低減を目的とし，特に地図生成において分割統治原理を応用して，依存関係が制限された小さなサイズの多くのサブマップを階層的に細分割・生成し，それぞれに対する最小二乗マップの最適化と近似を行う手法[3]を，Alexander Schiotka らは，モンテカルロローカリゼーション手法[4]を提案し，広い領域における SLAM 処理でもメモリ消費量を大幅に削減できることを示した．

これらのアプローチは，主に SLAM における地図生成アルゴリズムの改良により，メモリ消費量の削減を達成している．しかし，特定の SLAM のアルゴリズムのみしか考慮していないため，特定の SLAM アプリケーション改善しか実現できず，汎用性に課題がある．

このことから，本研究ではハードウェアリソースが限られた計算機での SLAM 負荷を汎用的に低減する方法として，近距離計算機であるエッジへのオフロードを検討するものとした．エッジ (Edge) とは，IoT デバイス等の高度な計算処理要求に対して，ユーザやセンサノードなどの近くに配置することができる 5G の無線設備を持つ分散型のサーバである．しかし，本技術を用いて，SLAM の負荷低減を効果的に行うための方法については，未だ十分研究がなされていない．

そこで本研究では，エッジに対して SLAM が使用しているメモリのオフロード処理による負荷軽減手法を提案する．メモリ上のデータをエッジサーバにリアルタイムにオフロードする場合，処理自体が負荷にならないよう，転送のタイミングを適切に決定する必要があるため，メモリとネットワークの利用状況に基づく予測モデルを作成した．また，これに加えて ROS (Robot Operating System)，分散システム向きのコンテナオーケストレーションシステムである Kubernetes を組み合わせたシステムの設計を行った．

2. メモリとネットワークに関する予測モデル

2.1 予測モデルの提案

先に述べた提案を実現するために，システムごとに異なるリソース利用の状況の予測モデルを作成し，オフロードのタイミングを予測に基づき決定するためのモデルが必要と考えた．そこで，次の式(1)で表されるメモリ使用率に関する予測モデルと式(2), (3), (4)で表されるネットワーク負荷予測モデルを定義した．メモリ予測にあたっては，ヒープ，スタック，バッファ，キャッシュメモリのデータ量を，ネットワーク負荷の予測にはシステム全体で送受信されているデータの量を把握するものとした．

$$M = a_1 \times M_{stack} + a_2 \times M_{heap} + a_3 \times M_{buf} + a_4 \times M_{cache} \cdots (1)$$

$$N_{load} = b_1 \times (N_{send} - \overline{N_{send}}) - b_2 \times (N_{receive} - \overline{N_{receive}}) \cdots (2)$$

$$b_1 = (N_{send} + N_{receive}) \times \frac{1}{\frac{N_{send}}{min(\overline{N_{send}}, \overline{N_{receive}})}} \cdots (3)$$

1 芝浦工業大学
Shibaura Institute of Technology
2 株式会社 Ales
Ales Inc.
3 東海大学
Tokai University

$$b_2 = (N_{send} + N_{receive}) \times \frac{1}{\frac{N_{receive}}{min(N_{send}, N_{receive})}} \cdots (4)$$

2.2 予測モデルの評価

本評価では CPU とシステム全体のメモリ利用率, ネットワーク送受信量が取得できる測定ツールを作成した. また, これの他にメモリ構成要素のデータ量について取得が可能な VALGRIND と VMSTAT を使用した.

まず，メモリ利用予測モデルの評価結果を図 1 に示す. 図 1 からは，どのデータセットにおいても，提案モデルと実値の誤差が 2%を切っており，実値に近い予測を出すことができていることが伺える.

次にネットワーク負荷予測モデルの評価結果を表 1 に示す．まず，N_{load}の上限/下限値に着目すると，上限/下限値が 0 をまたいで正負の値を取っていた．また，どのテストケースに対しても，ネットワーク送信の負荷が小さいと判定できるタイミングの出現確率$p(N_{load} \leq 0)$ は 5%以上であり，最大継続時間は 1 秒以上であるということが分かった．この結果から，最低でも平均約 3.3 秒に 1 回ネットワーク送信の影響が小さいと判断できるタイミングがあることが示された．このことから，オフロードを行うには十分な性能が確保でき，また予測モデルを用いること手法が有効である可能性が示された.

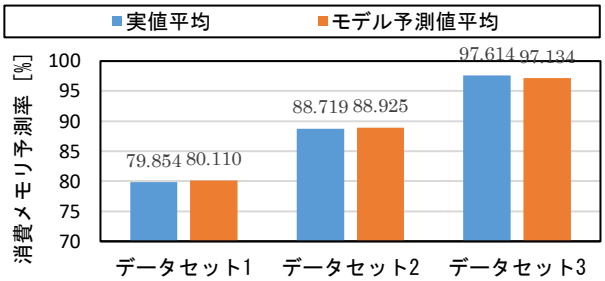

図 4　実値と予測値の平均の比較

表 5　データセットごとの
上限/下限値・$p(N_{load} \leq 0)$・最大継続時間

データセット	N_{load} 上限/下限	$p(N_{load} \leq 0)$	最大継続時間[s]
1	23.05/-64.14	22.22%	2
2	24.09/-13.48	14.29%	2
3	0.52/-1.55	5.56%	1

3. メモリオフロードシステムの設計

本研究では，実際にオフロード処理を実現するにあたって，ロボット向けのミドルウェアである ROS (Robot Operating System) と分散システム向きのコンテナオーケストレーションシステム Kubernetes を用いることとし，その設計を行った．実行の流れは次のようになる.

まず，システム全体に構築された Kubernetes のロボット側の，SLAM 処理関連ノードを含む実行プロセスを登録しておく．起動時にロボット側の登録プロセスをメモリ測定ツール (以下 meminfo) 上で動作させ，そのプロセス ID (PID) をリストや map などのデータ構造に記録する．ロボット側で実行している meminfo は取得したメモリ構成要素のデータ量を，ネットワーク通信量測定ツール (以下 netinfo) ではネットワーク送信量と受信量を，エッジに対し 1 秒間隔で ROS のメッセージとして送信する．エッジ上で，送られてきた統計データを一定時間ごとにモデル化し, 作成したパラメータをロボット側に ROS メッセージとして送信する．ロボット上で送られてきたパラメータを基に meminfo と netinfo で得られたデータを基に予測値を算出する．仮に予測値が閾値を超えており，かつネットワーク負荷が低いと判断できる場合は，PID を基にプロセスを停止させ，それによって得ることができる実行情報が含まれるコアダンプを，ROS を介してエッジに転送する．図 2 に実際の構成を示す.

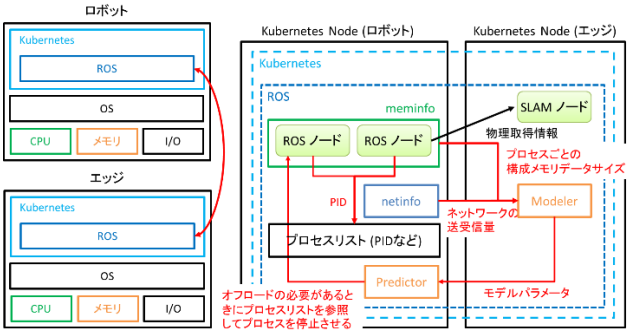

図 2　システム設計

4. まとめと今後の方針

本稿では，SLAM 処理に伴い多量に発生するメモリのオフロードを行うにあたり, 予測モデルの提案と評価を行い，これを用いた手法が汎用性の確認をはじめとする課題解決に十分有効であるという可能性を示した．また，これに加え ROS/ROS2 と Kubernetes を用いたシステムの設計を行った．今後はこのシステムの実現に向けて実装を行い，様々な環境による評価を行い，有効性を確認する方針である.

参考文献

[1] 文部科学省・経済産業省．“先端ロボット技術によるユニバーサル未来社会体験の実現に向けて”．平成 27 年版, 2015. https://www.kantei.go.jp/jp/singi/keizaisaisei/wg/kaikaku/dai6/siryou3.pdf (参照 2020-06-10)

[2] MathWorks. “SLAM とは？これだけは知っておきたい 3 つのこと”. https://jp.mathworks.com/discovery/slam.html (参照 2020-08-31)

[3] Benjamin Sugar. An approach to solving large-scale SLAM problems with a small memory footprint. 2014 IEEE ICRA, 2014. p. 3632-3637

[4] Alexander Schiotka. Robot localization with sparse scan-based maps. 2017 IEEE/RSJ IROS, 2017. p. 642-647