

モバイルロボット群向けミドルウェアの概要と 移動命令の microprotcol への適応

東原 大記[†], Sangsubhan Smath[†], Defago Xavier[†],

[†] 北陸先端科学技術大学院大学 情報科学研究科

モバイルロボットは、ハードウェアアーキテクチャやオペレーティングシステム、モバイルロボットの振る舞いを記述するためのプログラミング言語などが異なるヘテロな環境であるため、モバイルロボット向けのソフトウェア開発も複雑で膨大な量となる。既存のモバイルロボット向けミドルウェアは、モバイルロボット間の個体差を隠蔽する事はできるが、モバイルロボット間の協調的作業はモバイルロボットの利用者自身が記述する必要があった。本稿では、モバイルロボットの制御を行うミドルウェアの比較を行う。その後、microprotocolの概念を用いたモバイルロボットの協調的動作の基本的なサービスを提供するミドルウェアの基本的な設計に関して述べる。また、モバイルロボットの協調的動作において最も重要となるモバイルロボットの移動の制御を microprotocol に適応する。

Overview of middleware for group of mobile robots and adjustment of motion order to microprotcol architecture

Higashihara Daiki[†] Sangsubhan Smath[†] Defago Xavier[†]

[†] School of Information Science, Japan Advanced Institute of Science and Technology

The software development for mobile robots begins to large scale and complex, because the mobile robot environments are heterogeneous for hardware, operating system and programming language for describing mobile robot behaviors are different. Current middleware for mobile robots can hide individual difference. However the programmer of mobile robot cooperation needs to describe the behavior of mobile robot cooperation by themselves. In this paper, we compare the middleware for mobile robot controlling. And, we discuss about basic design of middleware that we will develop. The middleware will provide the basic services for mobile robots cooperation using the microprotocol software architecture. Finally, microprotocol software architecture meets motion controlling for mobile robot, because motion controlling for mobile robot is one of the most important element for mobile robot cooperation.

1 はじめに

近年、ロボット技術と計算機技術の進歩により、ロボットは高速に複雑な演算を行えるようになり、複雑な処理を行えるようになった。しかし、複雑な処理をロボットで行うためには複雑な機構を持つロボットを用いることになり、ロボットの開発にかかる時間的、金銭的成本が大きくなる。また、複雑な機構のロボット向けアプリケーションは、非常に大き

く複雑な物になる場合が多く、アプリケーションの信頼性を保つことも困難である。最低限の機能を持つ移動型のモバイルロボットを複数用いて協調的な作業を行う事で、大型で複雑な機構のモバイルロボットを用いて処理する場合と同様の処理が可能である。しかし、複数のモバイルロボットを用いて1つの処理を行う場合、モバイルロボット間の協調が重要となる。モバイルロボットの基本的な構成

は、駆動部、センサー部とその他のデバイスに分類できる。モバイルロボットの制御をするアプリケーションは、複数のセンサーからの情報を高速に処理し駆動部の制御をすることが重要であるため複雑になる。しかし、多くのロボットは、ロボットの開発者が独自にアプリケーションを開発しているため、ロボット毎に別のプログラムを書く必要がある。また、ロボットの個体差を隠蔽するためのミドルウェアが開発されている。既存のモバイルロボット向けのミドルウェアは、モバイルロボット毎の個体差を隠蔽し共通のロボットの制御アプリケーションを複数のアーキテクチャ上で実行することができる。しかし、モバイルロボット間の協調的な動作に関しては、モバイルロボットの利用者が独自に記述することが必要であった。Player/Stage¹⁾は、複数のモバイルロボットのセンサーを用いた動作をシミュレート可能なシミュレーターで、多くの研究で用いられている。また、モバイルロボットの協調動作には、既存の分散システムとの共通点が多い。既存のモバイルロボット向けのミドルウェアでは、モバイルロボット間での通信をサポートしていない場合や、モバイルロボット間の協調に関する処理はモバイルロボットの利用者が書く必要があった。Nekoは、分散アルゴリズムをシミュレートするためのフレームワークである。ソフトウェアアーキテクチャとして、microprotocolを用いている。しかし、モバイルロボットの重要点のひとつである移動に関してはサポートしていない。本稿では、モバイルロボット間の協調動作に関する基本的なサービスを提供するミドルウェアの開発を行うため、まず既存のソフトウェアの比較検討を行う。また、モバイルロボットの協調の基本である移動に関する microprotocol 間での通信方法に関して述べる。2章では、モバイルロボット向けのミドルウェアに求められるコンセプトに関して述べる。3章では、モバイルロボットの比較分類化を行う。4章では、ミドルウェアの基本的なデザインに関して述べる。5章では、モバイルロボットの移動の制御に microprotocol の考え方を適応する。6章では、本研究の今後の予定と課題に関して述

べる。

2 モバイルロボット向けミドルウェア

本章では、複数台のモバイルロボットが協調的な動作を提供するミドルウェアにおいて重要な概念に関して述べる。ソフトウェア開発において、ミドルウェアはソフトウェアの開発にかかる時間とプログラム記述にかかる仕事量を軽減する目的で提供される。ミドルウェアは、頻繁に利用される機能やサービス、データ構造等をフレームワークとして提供する。ミドルウェアで提供されるサービスやデータ構造は、十分な設計と設計に対する振る舞いが保証されている事が重要となる。特に、モバイルロボットは物理的な移動をする点が既存のミドルウェアと最も大きく異なる点であり、モバイルロボット向けのミドルウェアは既存のミドルウェアと比べ複雑なものになる。よって、サービスの振る舞いを保証するために、各サービスは最低限のサービス群に分割し各々のサービスは独立した振る舞いをする事が重要である。モバイルロボットは、各モバイルロボット毎にハードウェアアーキテクチャの構成やオペレーティングシステムなどが異なっているヘテロな環境である。既存の多くのモバイルロボット向けミドルウェアでは、モバイルロボット毎の個体差を意識すること無くモバイルロボットの利用者がモバイルロボットの制御アプリケーションを記述することができる。モバイルロボット間の協調的な動作に関しても、モバイルロボット向けのアプリケーション内には共通の部分がある。既存のモバイルロボット向けのミドルウェアにおいて、モバイルロボット間の協調動作を記述可能なミドルウェアは存在するが、モバイルロボットの利用者自身がすべて記述する必要があった。

2.1 再利用性

モバイルロボットの環境はヘテロな環境である事は前述の通りである。しかし、モバイルロボットの基本的な移動に関する制御のためのプログラムには共通な点が多い。既存の多くのモバイルロボット向けのミドルウェア等でも、モバイルロボットの制御をするアプ

リケーションが駆動系へ命令を共通のインタフェースとして提供する。また、センサーからのデータの収集に関しても、センサーの種類ごとにインターフェースを提供する。以上の単一のモバイルロボット内での命令と同様に、モバイルロボット間の協調作業に関する制御にも共通点がある。モバイルロボット間の協調的動作に関しても、最小の共通点ごとに分割しコンポーネントとして提供する事により、モバイルロボットの利用者はモバイルロボット間の協調に関するアプリケーションを容易に構築する事ができる。

2.2 独立性

一般的なソフトウェア開発において、ソフトウェアを構築する各コンポーネント間の関係は複雑ではなくお互いの依存関係が最小である事が重要である。特に、モバイルロボット向けのソフトウェアにおいては、モバイルロボットは物理的な移動を制御が重要であるため、各コンポーネント間の関係やコンポーネントの組み合わせによる副作用が最小限にすることによりモバイルロボットの制御を行うことにより、コンポーネント間の独立性を保つことが重要である。

2.3 信頼性

モバイルロボットは物理的に移動する。ソフトウェアの複雑なると、コンポーネント間の依存関係が大きくなる可能性があり、コンポーネントの組み合わせに副作用が発生する可能性がある。コンポーネントの組み合わせに副作用が発生すると、ソフトウェア開発者の意図しない振る舞いをモバイルロボット行う。開発者の意図しないモバイルロボットの振る舞いは、モバイルロボット間の協調的作業を行う事が困難である。また、ソフトウェアを構成する各コンポーネントは、各々のコンポーネントの振る舞いが確実に保証される事が重要である。

2.4 応答時間

モバイルロボットは、センサーの情報を随時収集し情報を高速に解析する事により、安全に移動したりその他の振る舞いを行うことができる。コンポーネント間でセンサーの情報

の受け渡しや移動に関する命令に多くの時間を要した場合には、情報の解析を行う前に物理的な環境の状態が変更する場合があります。モバイルロボットは他のモバイルロボットや障害物に衝突したり、要求以上の移動距離を移動する事となる。

3 ソフトウェアの分類化

本章では、ソフトウェアの分類化を行い、各分類の特徴に関して述べる。既存のモバイルロボット向けのミドルウェアの多くは、モバイルロボットの移動を制御を行う。しかし、モバイルロボット間の協調的動作はモバイルロボット間の通信が重要な要素となる。

3.1 移動制御ベース

既存の多くのモバイルロボット制御のためのソフトウェアは、モバイルロボットの移動の制御に注目している。Player/Stageは、センサーを用いたモバイルロボットの制御を行うミドルウェアである。一般的に用いられるセンサーに関するインタフェースが用意されており、モバイルロボットの利用者が利用したいセンサーのインタフェースや、インタフェースを用いて提供されている実装などを用いることでセンサーの情報を利用するモバイルロボットのアプリケーションを実装する事ができる。Playerは、モバイルロボット上で実行するモバイルロボットを制御するためのネットワークサーバである。駆動系とセンサーの基本的な操作をIPネットワークを経由するインタフェースとして提供する。モバイルロボットの利用者は、TCPソケットを用いたクライアントプログラム上でセンサーデータの取得や駆動系の制御を記述する。また、Stageなどのシミュレータプログラムをモバイルロボットの代わりに、Playerサーバに接続することにより、実際のモバイルロボット上で実行するクライアントプログラムを用いてクライアントプログラムの振る舞いをシミュレートする事ができる。Player/Stageは、モバイルロボット間の通信をサポートしていないため、複数台のモバイルロボット間で通信を用いた協調に作業を行うためにはモバイルロボットの利用者自身で記述する必要がある。

3.2 コミュニケーションベース

Neko は、分散アルゴリズムを構築するための java フレームワークである。ソフトウェアアーキテクチャとして microprotocol を採用している。neko フレームワークを用いて構築した分散アルゴリズムの実装は、実際のネットワーク上と neko が提供するシミュレータ上の両方で同一のプログラムで実行する事ができる。分散アルゴリズムの構築を目的としたフレームワークであるため、モバイルロボットの移動の概念などはサポートしていない。

3.2.1 Microprotocol

Microprotocol^{2) 3) 4)} とは、フレームワーク内で組織化されたユニットのプロトコルコードであり、ソフトウェアコンポーネントである。Microprotocol では、他の microprotocol に対してよく利用される基本的なメカニズムを提供するサービスである。各 microprotocol の開発者と microprotocol の構築者が同じとは限らない。他の microprotocol の開発者が記述した microprotocol の実装の変更や実装の中身を構築者が熟知すること無く構築者はソフトウェアを構築する事ができる。Microprotocol を実装する際には、microprotocol 間の相互作用を最小限にすることが重要である。Microprotocol 間の相互作用を最小限にすることで、microprotocol のプログラムと microprotocol どうしの結合を完全に分離する事ができる。モバイルロボットの制御に必要である各メカニズムを提供する microprotocol を記述する開発者と構築者が同一人物であるとは限らないため、microprotocol のプログラムと結合の分離が可能な点がモバイルロボット向けのソフトウェアにおいても有用である。

4 ミドルウェアのデザイン

本章では、我々が提案し開発するモバイルロボット間の基本的な協調的動作をサポートするためのミドルウェアの基本的なデザインに関して述べる。

我々は、数台から数十万台までの、モバイルロボット群の協調作業が実現可能なミドルウェアの開発を目的とする。本ミドルウェアを用いる事により、モバイルロボット向けの

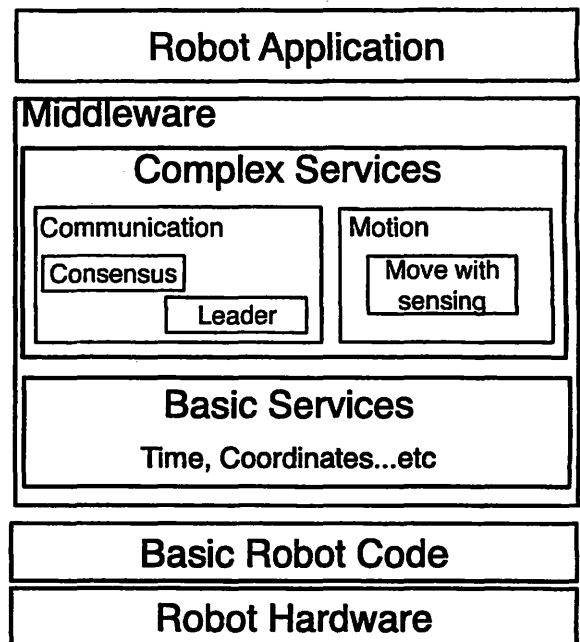


Fig. 1 モバイルロボットの協調的動作の基本的なサービスを提供するミドルウェアの概要

アプリケーションのプログラマは、1つのモバイルロボットシステムとして提供可能とする。また、モバイルロボットは物理的な作業をするため故障する可能性があるが、いくつかのモバイルロボットが故障したとしても、システム全体に影響する事の無いシステムの開発を目的とする。

Fig. 1 では、我々が提案するミドルウェアの概要を示す。

図の最下部をハードウェア (Robot Hardware)、最上部をモバイルロボットのアプリケーション (Robot Application) とする。既存の多くのミドルウェアでは、モバイルロボットのハードウェアやオペレーティングシステムの異なりを隠蔽する (BasicRobotCode)。本ミドルウェアは、個体差を隠蔽する BasicRobotCode として Player/Stage 上に構築する。モバイルロボット間の協調的動作のメカニズムを提供するミドルウェアに関しても、個々のモバイルロボットの異なりを意識する事無く利用可能とするために既存のモバイルロボット向けミドルウェアとモバイルロボットのアプリケーションの間で利用する。

モバイルロボット間の基本的な協調的動作を提供するミドルウェアは、協調的作業において重要となる基本的なサービス (Basic Service) と、基本的なサービスを利用して実際のモバイルロボット間での協調と協調に基づいた移動の制御を行う複雑なサービス群 (Complex Service) を提供する。基本的なサービスには、共通の時間や座標系等の基本的なメカニズムを提供する。複雑なサービス群は、microprotocol の考え方をを用いて記述される。複雑なサービス群には、通信を基本としたメカニズムとモバイルロボットの移動を基本としたメカニズムに分類すると考えられる。通信を基本としたメカニズムは、既存の分散システムと共通点の多い consensus や leader の選出アルゴリズムなどが考えられる。また、移動を基本としたメカニズムには、センサーを用いて周囲の環境を把握しながら移動するなどのアルゴリズムなどが考えられる。モバイルロボットのアプリケーションの構築者は、ミドルウェアで提供される microprotocol や他の開発者が記述した microprotocol を用いてアプリケーションを構築する。

5 モバイルロボットの移動制御を microprotocol への適応と構築

本章では、モバイルロボットの移動の制御を microprotocol へ適応する。また、モバイルロボットの移動制御に関する microprotocol を用いてモバイルロボットの協調に関するアプリケーションの構築に関して述べる。

5.1 移動制御を microprotocol へ適応

モバイルロボットの制御は、駆動系とセンサーを適切に制御する事が重要である。センサーには、GPS などのモバイルロボットの位置情報を取得するためのセンサーや赤外線や超音波等のモバイルロボットの周辺の環境を見るための視覚情報を取得するためのセンサーに大別する事ができる。しかし、位置情報を取得するためのセンサーも視覚情報を取得するためのセンサーも利用方法は、情報を取得し解析する作業である。以上の作業は、通信のために用いるメッセージの受け渡し作業に類似している。以上の事から、センサーのデータ受

座標系	移動制御
相対座標系	角度
相対座標系	速度と角度
絶対座標系	移動の範囲
絶対座標系	移動のルート
絶対座標系	移動の始点と終点

Table 1 座標系とモバイルロボットの移動命令の種類の組み合わせ

け渡しは、neko の既存のネットワークのメッセージ受け渡しと同じ様に扱う事ができる。

一方で、microprotocol 上でモバイルロボットの駆動系の制御する事に関しては、センサーのデータ受け渡しに比べ、モバイルロボットは物理的な移動が伴うため次の2つの問題点がある。

- ・移動命令の種類
- ・座標系の違い

Table 1 は、モバイルロボットの座標系と移動命令の種類の組み合わせを示す。

モバイルロボットの座標系には、絶対座標系と相対座標系がある。絶対座標とは、全てのモバイルロボットが共通の基準点を共有する座標系である。複数台のモバイルロボット間で協調的な動作を行いたい場合には、有用な座標系である。一方で、相対座標系とは、各モバイルロボットが各々に基準点を保有し、多くの場合各モバイルロボットの中心点が用いられる。モバイルロボットの駆動系は、絶対座標系を解釈する事が困難であるため、駆動系に直接命令する場合等には、有用な座標系である。モバイルロボット間の協調動作では、モバイルロボット間での通信が重要となるため、相対座標系を用いることはできない。モバイルロボットの移動の制御方法は、主に相対座標系を基準とする命令と絶対座標系を基準とする種類に分類することができる。相対座標系を用いた移動の制御は、モバイルロボットがセンサーから得たまわりの情報を考慮しながら移動する場合等に用いられる。一方で、絶対座標系では、モバイルロボット間のコミュニケーションを用いた協調動作を制御するために利用す

る場合が多い。モバイルロボット間の協調的動作を含むモバイルロボットの移動に関する制御は、モバイルロボット間の協調的動作が優先的に行われることが重要である。モバイルロボット間の協調的動作が優先的に行われるとは、モバイルロボットの移動を制御する microprotocol の組み合わせにおいて、モバイルロボット間の協調に関する micoroprotocol は、micoroprotocol の構成のユーザアプリケーション側に位置する事が重要である。

5.2 Microprotocol の構築

Microprotocol を用いてモバイルロボットの移動を制御するためには、microprotocol 間の相互作用を考慮することが重要である。Microprotocol の構築方法によっては、microprotocol の構成内に並行処理が発生する場合がある。モバイルロボットの移動制御においては、micoroprotocol の組み合わせによって発生する並行処理による相互作用がコンポーザの意図しない実行を引き起こすため、移動の制御には microprotocol をネットワークコミュニケーションに用いる以上に相互作用を考慮する事が重要である。

Microprotocol の相互作用を最小限にしモバイルロボットの移動を制御するためには、micoroprotocol への入力と入力に対する動作を適切に処理することが重要である。Microprotocol の相互作用を最小限にするためには、入力を受ける microprotocol が全ての入力を受信した後に処理を実行する。

6 まとめ

本稿では、モバイルロボット向けの中ドルウェアの比較とモバイルロボットの移動の概念を micoroprotocol の考え方に適応した。モバイルロボットの移動に microprotocol の概念を適応するためには、microprotocol の相互作用を考慮する事が重要である。モバイルロボットの移動を制御するための microprotocol 間の相互作用を最小にするために、移動制御のメッセージを受信する microprotocol は全ての入力を受けた後で処理をする。モバイルロボットの協調的動作は複雑である。モバイルロボットの協調的動作から、基本的なサービスとし

て提供する事が重要な要素と基本的なサービスを用いた複雑なサービス群に分離する事が今後の課題である。また、ミドルウェアが提供するサービス群を用いて、実際のモバイルロボットアプリケーションを記述するためのインターフェースの分類と提供も重要な課題である。

参考文献

- 1) Brian Gerkey, Richard T. Vaughan and Andrew Howard. "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems". In Proceedings of the 11th International Conference on Advanced Robotics (ICAR 2003), pages 317-323, Coimbra, Portugal, June 2003.
- 2) Sergio Mena, Xavier Cuvellier, Christophe Grégoire, André Schiper, Appia vs. Cactus: Comparing Protocol Composition Frameworks, 22nd International Symposium on Reliable Distributed Systems, 189-198, 2003.
- 3) N. T. Bhatti, M. A. Hiltunen, R. D. Schlichting, and W. Chiu, Coyote: A system for constructing negrain congrable communication services. ACM Transactions on Computer Systems, 16(4), 321-366, Nov. 1998.
- 4) M. A. Hiltunen and R. D. Schlichting, A congrable member ship service, IEEE Transactions on Computers, 47(5), 573-586, 1998.
- 5) Péter Urbán, Sergio Mena, Xavier Défago and Takuya Katayama. Concurrency in microprotocol frameworks. Research Report IS-RR-2006-004, Japan Advanced Institute of Science and Technology, Ishikawa, Japan, February 2006.